PORTFOLIO

Charles D Rady

Table of Contents

Section I — Resume Materials

Resume	2
Tech skills	3
Cover letter	4

Section II — Systems Analysis & Design

Elaboration Phase	5
SalesForce user manual	85
Class Diagram from iteration 5	97
ERD from iteration 5(CIS320)	98

Section III — Object-Oriented Programming

Library version 2 code	99
GradeBook code	127
Linq exercise	133
Sort exercise	137

Section IV — SQL: Database Design

Queries (assignment 8, CIS 320)	140
Star Schema/Data Warehouse	146
ERD with business rules	152

Section V — Internet (Web Site Design)

PHP code	154
HTML code	163

Section VI — Technical Writing

AT&T Goatse iPad incident	165
Standards & Practices/ Non-Profits	171
Case report: Waco Manufacturing	177

Appendix A — Library Class Heirarchy

Class Hierarchy for Library	182

Appendix B — Legacy Systems (DOS)

Legacy: DOS batch file	201

Appendix C — cdradyproductions.com

203

Appendix D — References

204

CHARLES RADY

8009 WANDA WAY
LOUISVILLE, KY 40219
(502)774-0115
CHUCK@CDRADYPRODUCTIONS.COM
CDRADYPRODUCTIONS.COM

EDUCATION

University of Louisville College of Business
Bachelor of Science in Business Administration
Major: CIS expected Fall 2011

Eastern Connecticut State University
Music Performance major
Accepted into a full time music performance job directly out of school.
-Remained successful in this career for 10 years

HONORS

At the University of Louisville
DEAN'S LIST: FALL 2008, FALL 2009, SPRING 2010, SUMMER 2010, FALL 2010
Tau Sigma National Honor Society for Transfer Students
Inducted Fall of 2009
At Eastern Connecticut State University
Dean's List

EXPERIENCE

WEST LOUISVILLE MINISTRIES
PRESENT
Implementing SalesForce NPSP database
with custom-developed objects & relationships

HAND IN HAND MINISTRIES
Fall 2010
Consulting (design)
SalesForce transition from eTapestry to NPSP
Normalization
Database Design w/ Visio
Process redesign

HOLLAND AMERICA LINE — 2007-2008
Seagoing performer for Holland America Line for two years.
Ships sailed: M.S. Maasdam, M.S. Volendam, M.S. Westerdam.

OPPORTUNITY HOUSING INCORPORATED 2005-2006
Mental Health Rehabilitation Technician and Certified Residential Medication Aide in Bangor, Maine.
Duties: Direct supervision of patients as well as administering timed and requested doses of schedule II medication.
Responsible for 8 people and over $20,000 of controlled medications at any one time.

UNIVERSAL STUDIOS FLORIDA — 1998-2004
Dueling piano bar performer in an all-request piano bar located in the CityWalk section of Universal Studios
Direct Supervisor: Justin Bowen (407) 224-2698

Technical Skills

Programming Languages
C#
HTML
SQL
PHP
BASIC

Applications
Visual Studio
MS Office Publisher
MS Office Word
MS Office PowerPoint
MS Office Excel
MS Office Project
MS Office Visio
MS Expression Web 4
MS Expression Blend 4
MS Expression Media 2
MS Expression Encoder
MS Expression Design 4
SQL Server
Logic
Cubase
iSuite
WireShark
SoqlXplorer
Aperture
Photoshop
Various video editing
Various DVD authoring
MS SharePoint
Open Office
GoogleDocs
VST (all)
Disk Utilities (all)

Databases
SQL Server
MS Access
SalesForce CRM

Operating Systems
Windows 95, 98, ME, XP,
Vista, 7
Mac OS Legacy
Mac OS X
"Tiger"
"Leopard"
"Snow Leopard"
iPhone/iPad OS
MS DOS
LINUX
BIOS

Networking
Cisco Routers
D-Link Routers
Apple Routers
Linksys Routers
Network Monitoring
Media Center Integration
Network Security
Network Configuration

Technical Writing
Research
Case Reports
UML 2.0
Instruction Manuals

Physical
Custom PC's
PC repair/upgrades
MacBook repair/upgrades

SAMPLE COVER LETTER

Charles Rady
8009 Wanda Way
Louisville, KY 40219
March 16, 2011

Dear: HR Representative

I am writing this short letter to express my interest in taking part in your (up to) 6 credit hour co-op program. I am a University of Louisville College of Business senior in the traditional CIS program. I am also a non-traditional student in my thirties who has worked a ten-year career in the music industry before returning to school for a four-year degree. I feel we have much to offer each other and look forward to speaking with you at your convenience. If you wish to learn more about me I have been building a website at cdradyproductions.com. This website showcases my musical career along with a CIS section as well as some other sections I have been working on.

Thank you for your consideration,

Sincerely,

Charles D. Rady

This sample letter was written for the following job description:

Job Description:

Humana MIS co-op:
Obtain an internship to gain a greater understanding of Humana's Information Systems environment, tools, methods and techniques used to code, test, and implement computer solutions and develop technical skills. Assisting with technical support for all HMHS. Participating in group projects within the MIS department. Web page design/enhancement. Tasks include trouble shooting software, hardware, networks and printers. Installation of new software and hardware. Creating requests for new users to have network access and moving equipment.

ELABORATION PHASE

Group Project:
Charles Rady
Matt Nasca
David Mattingly
CK Darvin

Contents

Hand in Hand Ministries CRM ..... **Error! Bookmark not defined.**
Vision (Small Project) ..... **Error! Bookmark not defined.**
Version 1.0 ..... **Error! Bookmark not defined.**
Revision History 10
Introduction 11
Positioning 12
Problem Statement 12
Product Position Statement 13
Stakeholder and User Descriptions 14
Stakeholder Summary 14
Summary of Key Stakeholder or User Needs 15
Reasons for the problems - 15
How is it solved? 15
What solutions does the stakeholder or user want? 15
Alternatives and Competition 15
Product Overview 15
Product Perspective 15
Assumptions and Dependencies 16
Product Features 16
Gifts in Kind 16
Monetary Donations 16
Multiple Donations 16
Donor Database 16
Other Product Requirements 16
Feasibility Analysis 17
Technical Feasibility: Can we build it? 17
Familiarity with the Functional Area: Strong 17
Familiarity with the Technology: Strong 17
Size of the Project: Small 17
Compatibility: Strong 17
Conclusion 17
Economic Feasibility: Should we build it? 18
Benefits: 18
Costs 18

Feasibility Spreadsheet....19
Benefits Analysis....20
Cost Analysis....21
Organizational Feasibility Analysis: If We Build it Will They Come?....22
Strategic Alignment:....22
Stakeholder Analysis:....22
Conclusion:....22
Requirements....23
Review and Approval Signatures....23
1. Scope....24
2. Overview and Objective....24
3. Business Justification....24
4. System Requirements....24
4.1 Functional Requirements....24
1. Monetary Donations....24
2. Non-Monetary Donations....24
3. Donor Information....24
4. Campaigns....24
5. Wish List....24
6. Volunteer Information....25
4.2 Non-Functional Requirements....25
1. Operational Requirements....25
2. Performance Requirements....25
3. Hardware Requirements....25
4. Software Requirements....25
5. Processes to Be Automated....26
Use Case Descriptions....28
Accept Monetary Donation....28
Accept Gift in Kind Donation....29
Manage Contacts....30
Create Mailing List....31
Manage Wish List....32
Manage Gift-In-Kind Inventory....33
Track Local Volunteer....34
Track Immersion Trip Volunteer....35

Use Case Diagram....36
Use Case Narrative....36
Create Mailing List....37
Track Local Volunteer....38
Track Immersion Trip Volunteer....39
Manage Gift In Kind....40
Manage Contact....41
Manage Wish List....42
Accept In Kind Donation....43
....43
Accept Monetary Donation....44
Class Diagram....45
Database Design....46
Screen Layouts....48
Main Menu....48
Navigation Screens....49
Manage Contact....49
....50
....50
Accept Gift In Kind....50
Accept Monetary Donation....51
Contact Search Results....52
Create Mailing List....53
Edit Inventory Item....54
E-Mail....55
History....56
Manage In Kind Inventory....57
Manage Wish List....58
Merge Contact....59
To Do / Tasks....60
Track Immersion Trip Volunteer....61
Track Local Volunteer....62
Calendar....63
Windows Navigation Diagram....64
Deployment Diagram....65

Nonfunctional Requirements....66
Nonfunctional Requirements....67
1. Operational Requirements....67
2. Performance Requirements....67
3. Hardware Requirements....67
4. Software Requirements....67
5. Security Requirements....68
Hand in Hand Security Policy....68
Salesforce Security Policy....68
Prototype....69

Revision History

Date	Version	Description	Author
09/27/2010	1.0 Revision 1.1	Reformatted, added categories to product features, fixed minor errors, spelling	David Mattingly, CIS 320 group Wednesday
10/12/2010	1.0 Revision 2.0	Extended product features, added economic feasibility analysis	David Mattingly, CIS 320 group Wednesday
10/20/2010	1.0 Revision 2.1	Revised minor errors, Added organizational and technical feasibility, Major Revision of economic feasibility	CIS 320 Group Wednesday

Introduction

The client, Steve Croghan, who represents the non-profit organization Hand in Hand Ministries, seeks a better solution for customer relationship management (CRM). More specifically, Mr. Croghan would like to evaluate how the current application that he and his colleagues are using, which is a CRM program called eTapestry, measures up to other available CRM's (particularly Salesforce's Non-Profit Starter Pack). eTapestry's inability to perform task like automatic gift giving/donations that are scheduled on a predetermined basis (biweekly, bimonthly, annually, etc.). Another problem was not having the ability to produce receipts or add new donors to Hand in Hand Ministries contact list. Other complications that Hand in Hand Ministries would like to be addressed are thing such as determining prices for products that are being donated, managing existing contacts (updating addresses and other features), and the ability to construct a routine system for monthly/quarterly newsletters and other mailing items.

Positioning

Problem Statement

The problem of	eTapestry not providing needed features.
affects	• Hand in Hand Ministries • Donors • End Users, Staff • Volunteers
the impact of which is	Hand in Hand Ministries systems are inefficient and time consuming. eTapestry lacks features and is expensive.
a successful solution would be	Able to perform features that eTapestry cannot. See section 4.1 “Product Perspective” <u>Automated gift giving</u> • Fund and manage multi-year commitment (Legacy) donations • Provide a system for recurring donations • Automatically bill donors based on planned donations <u>Gifts in Kind</u> • Automate Gifts in Kind donations • Track/Manage Gifts in Kind

Product Position Statement

For	Hand in Hand Ministries
Who	need a better CRM to manage their donor database.
The (product name)	is a Customer Relationship Management
That	Automated Gift Giving, Automated Receipts, emailing system (including newsletter issues)
Unlike	eTapestry
Our product	Will correct the issues that Hand in Hand Ministries is experiencing.

Stakeholder and User Descriptions

Stakeholder Summary

Name	Description	Responsibilities
Donors	Person who makes a gift of property	Supplies Funding, or Gifts in Kind
Volunteers	Offer work/services for free	Collect gifts in kind or other donations
Users/Staff	End User	Produces Reports Manages Donations Provide donors with receipts for gifts-in-kind
Organizational Management	Managers/Board of Directors	Budget enough money for project Encourage users to accept the new system
Champion	Steve Croghan	Initiates the Project Promotes the Project Provides Recourses Allocates time to the Project

Summary of Key Stakeholder or User Needs

Reasons for the problems -

eTapestry does not support the features that Hand in Hand Ministries needs to perform tasks like automatic gift giving/donations that are scheduled on a predetermined basis (biweekly, bimonthly, annually, etc.). Gifts In Kind are manually entered because there is no automated system or process. These problems are stated in the introduction.

How is it solved?

By providing Hand in Hand Ministries with new applications that can provide features that eTapestry cannot. Automating the Gifts in Kind process. The solution is stated in the **Product Overview**. In addition, the impact of the new system is stated by user in the **Organizational Feasibility Analysis**.

What solutions does the stakeholder or user want?

Aside from fixing the problems that were previously mentioned, Mr. Croghan was also in search of a solution pertaining specifically to eTapestry's cost, describing how the removal of that price would prove to be quite beneficial to Hand in Hand Ministries.

Alternatives and Competition

There is a wide array of nonprofit CRM software available on the market. Hand in Hang ministries is not ruling out any possible software as long as their needs are met. Building a homegrown solution is not recommended for HHM as the technological resources in house are not there.

Product Overview

Product Perspective

This application (or applications) would solve the issues with automatic gift giving/donations, emailing system (including newsletter issues) and receipts. To solve the issue with determining prices, our associates came up with the idea of using a mobile device (smartphone or portable computing device) to capture a products UPC and then present the user with the products suggested price(s). Items that do not have a suggested retail price can be viewed on an existing document that contains a list of products that are related, keeping the concerns of condition in mind; with use of the device (portable computing devices can perform this task as well). As for adding new clients (in immediate manner) to a contact list, can be done with use of the same device used for scanning the products UPC (A portable computing device would work relatively well in this example). The product will also be able to migrate data from existing eTapestry database.

Assumptions and Dependencies

This project assumes that when a Gift in Kind is donated there will be either a portable device or computer available to enter the donation electronically. If this device is not present, the Vision document and deliverables will change.

Product Features

Gifts in Kind

- Accept and track gifts-in-kind inventory
- Provide donor receipts for gifts-in-kind
- Provide an ability to record, track, and price “gifts in kind” electronically

Monetary Donations

- Accept and track Monetary donations
- Accept payment from credit card and e-check accounts

Multiple Donations

- Fund and manage multi-year commitment (Legacy) donations
- Provide a system for recurring donations
- Automatically bill donors based on planned donations

Donor Database

- Improve queries for the donor database
- Integrate a customer relationship management system for marketing and newsletter campaigns
- Integrate the database with the (NOCA) National Change of Address service
- Merge contacts within the customer relationship management system
- Migrate current donor database
- Provide documentation of transactions automatically.

Other Product Requirements

- Hand in Hand Ministries will either provide or purchase hardware needed for project.
- End Users will need to be trained on how to use new applications
- The Data must be migrated into the new system from eTapestry
- The system will be compatible with software that is currently in use (i.e. Microsoft Excel)

Feasibility Analysis

Technical Feasibility: Can we build it?

This analysis will determine the extent to which the system can be successfully designed, developed, and installed by the IT Group.

Familiarity with the Functional Area: Strong

The new system will incorporate Salesforce, which is a preexisting Customer Relationship Management software for nonprofit organizations. This system will be easier to build and implement than a homegrown solution for several reasons.

1. The users are already familiar with another CRM (eTapestry).

2. In general, the development of new systems is riskier than extensions to an existing system because existing systems tend to be better understood.

3. We as analysts are new to the business functional area so using a prebuilt CRM is ideal.

Familiarity with the Technology: Strong

Salesforce maybe not the same as eTapestry but both are technologically similar. The users should be familiar with the technology even though they have not used Salesforce before. The training should be fairly easy and short which will keep problems and delays to a minimum.

Size of the Project: Small

The size of the project is relatively small. The development team consists of four IT students. And the number of end-users is eight.

Compatibility: Strong

The new system will be integrated into the existing environment and use software and technology that Hand in Hand Ministries already uses (See **Product Features** in Vision Document).

Conclusion

Implementing the system is very feasible. Hand in Hand Ministries is already familiar with the functional area and technology that the system uses. The project is small which means less risk. And the system is compatible with the preexisting systems of Hand in Hand Ministries.

Economic Feasibility: Should we build it?

Benefits:

The system is being created primarily to save time and money. The system alone will save Hand in Hand ministries up to $4,500 per year in licensing fees. Although much of the system will be intangible benefits including improved functionality, reducing redundant processes of eTapestry, and easier donor management. Economically the system justifies being built because the benefits outweigh the cost as seen in the Feasibility Analysis. If the system increases the total donations received by .5 to 2 percent each year, that's $15,000 to $60,000 more donations annually. (These numbers are based on the 2.9 million raised by Hand in Hand ministries last year.) With better donor management and donation processing this is a very possible result. The Feasibility analysis assumes that each year the donations increase by 1.25% (This is the average of the expected increase of donations).

Costs

Development Costs

The analysis assumes that a team of four members will implement the system at $20 per hour which comes out to $64,000 (4 people * 40 hours/week * 4 weeks/month * 5 months * 20$ per hour). Not many supplies will be needed apart from what Hand in Hand Ministries is already providing. The new system will need a scanner and computer or portable device for tracking gift in kind donations. Supplies are estimated to be around $5,000.

Operational Costs

For support one of the team members can be kept on retainer for the duration they are needed. The first year the cost is estimated to be at $15,600. This is (52 weeks * 15 hours/week * 20$ per/hour. Each year the support cost will decrease due to increased familiarity of the system.

Return:

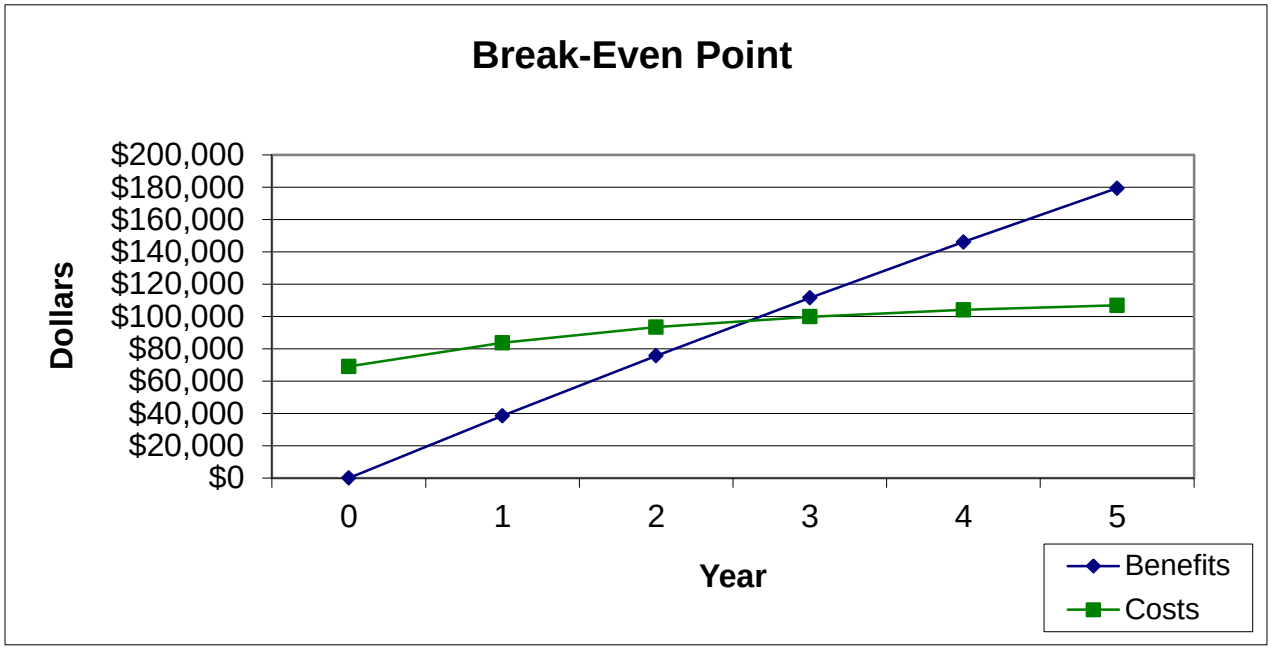
Return on Investment	- 67.8%
Break Even Point	- 2.62 Years
Internal Rate of Return	- 28.4%

Conclusion:

Economically the system is sound. After 2.62 years the system is expected to break even. The system should be built.

Feasibility Spreadsheet

Cost Avoidance		$4,500	$4,500	$4,500	$4,500	$4,500	$22,500
Total Benefits	$0	$40,832	$41,740	$42,671	$43,625	$44,603	$213,472
PV of Benefits	$0	$38,521	$37,149	$35,827	$34,555	$33,330	$179,382
NPV of Benefits	$0	$38,521	$75,669	$111,497	$146,052	$179,382	
Development Costs							
Personnel	$64,000						$64,000
HW/SW/Supplies	$5,000						$5,000
Development Costs	$69,000	$0	$0	$0	$0	$0	$69,000
Operational Costs							
Personnel		$15,600	$10,920	$7,644	$5,351	$3,746	$43,260
HW/SW/Supplies		$0	$0	$0	$0	$0	$0
Operational Costs	$0	$15,600	$10,920	$7,644	$5,351	$3,746	$43,260
Total Costs	$69,000	$15,600	$10,920	$7,644	$5,351	$3,746	$112,260
PV of Costs	$69,000	$14,717	$9,719	$6,418	$4,238	$2,799	$106,891
NPV of Costs	$69,000	$83,717	$93,436	$99,854	$104,092	$106,891	
Cash Flow							
Yearly NPV Cash Flow	-$69,000	$23,804	$27,430	$29,409	$30,317	$30,531	$72,491
Overall NPV Cash Flow	-$69,000	-$45,196	-$17,767	$11,643	$41,960	$72,491	
Performance							
ROI							67.8%
Break Even Point							2.62
IRR							28.4%
Assumptions							
Interest Rate	0.06						
Discount Rate	1.000	0.943	0.890	0.840	0.792	0.747	



Benefits Analysis

	Year 0	Year 1	Year 2	Year 3	Year 4	Year 5	Total
Increased Revenues							
Increased Donations		$36,332	$37,240	$38,171	$39,125	$40,103	$190,972
							$0
							$0
							$0
Total Revenues		$36,332	$37,240	$38,171	$39,125	$40,103	$190,972
Cost Avoidance							
eTapestry		$4,500.00	$4,500.00	$4,500.00	$4,500.00	$4,500.00	
Total Savings		$4,500	$4,500	$4,500	$4,500	$4,500	$22,500

Cost Analysis

	Year 0	Year 1	Year 2	Year 3	Year 4	Year 5	Total
Development Costs							
Personnel	$64,000.00						
HW/SW/Supplies	$5,000.00						
Total Development Costs	$69,000.00						
Operational Costs							
Personnel		$15,600	$10,920	$7,644	$5,351	$3,746	$43,260
HW/SW/Supplies							
Total Operational Costs		$15,600	$10,920	$7,644	$5,351	$3,746	$43,260

Organizational Feasibility Analysis: If We Build it Will They Come?

This analysis will try to determine how well the system will be accepted by the end users at Hand in Hand Ministries and incorporated into the ongoing operations of the organization.

Strategic Alignment:

The project is a strong fit with the goals of Hand and Hand Ministries. The system request states "Hand in Hand Ministries is interested in migrating from their current CRM provider (eTapestry) to a more cost effective solution. They would also like to improve their current business practices to reduce redundancy as well as streamline and integrate their donor management efforts with a system that can automate many of their currently manual processes." Implementing the new system will not only reduce costs, but also integrate many of Hand in Hand's processes.

Stakeholder Analysis:

Name	Description	Impact of New System
Donors	Person who makes a gift of property	Improved service
Volunteers	Offer work/services for free	Easier collecting of Gifts in Kind
Users/Staff	End User	Easier Reporting Better donor and donation Management
Organizational Management	Managers/Board of Directors	Increased amount of donations Less money spent on a better CRM Staff and Volunteers will be happier with new system.

Conclusion:

The new system is a good fit for Hand in Hand Ministries. The system will provide a more cost effective solution than eTapestry and automate many redundant features of current processes. The system will also have a positive impact on all stakeholders.

Requirements

Hand in Hand
Ministries
System
Version: 1.0

AUTHOR:

The author's signature indicates that this document was written for the named system to meet the [Company Name] Quality Standard for Computer Validation requirements for system documentation.

Signature	Initials	Date	Unit

Charles Rady

Review and Approval Signatures

The signatures below indicate that this document meets Hand in Hand Ministries requirements for documenting business area requirements, while accurately reflecting the business area functional requirements for the named system.

REVIEWED BY TEAM MEMBER

Signature	Initials	Date

Matt Nasca

APPROVED BY TEAM MEMBER

Signature	Initials	Date

CK Darvin

APPROVED BY TEAM MEMBER

Signature	Initials	Date

David Mattingly

APPROVED BY SYSTEM OWNER OR DESIGNEE

Signature	Initials	Date

Steve Croghan

1. Scope

This document establishes the Hand in Hand Ministries donor tracking & processing business and technical requirements for the NPSP migration from eTapestry.

2. Overview and Objective

We intend to move Hand in Hand Ministries' data from eTapestry to Salesforce NPSP and make that data easier to organize and query. In the process of changing systems, we also intend to streamline and combine processes for a more efficient user experience. When the new system is operational, we hope to have included forms and processes for dealing with key areas that Hand in Hand has made apparent to us such as Gift in Kind and legacy donation processing.

3. Business Justification

This system is being created primarily to save time and money. Using NPSP will save up to $4,500 per year in licensing fees alone. By automating currently redundant processes we can eliminate unnecessary steps and save Hand in Hand much of the effort it currently puts into running their eTapestry-based donor management system.

4. System Requirements

The following describes the functional and non-functional requirements of the new system proposed for Hand in Hand Ministries. Due to the web-based nature of this solution hardware requirements are at a minimum. This is because most of the hardware at Hand in Hand will be used as clients interacting with the main system located at and maintained by Salesforce.

4.1 Functional Requirements

1. Monetary Donations

1.1 The system shall accept monetary donations.
1.2 The system shall provide documentation of transactions automatically.
1.3 The system shall accept and manage multi-year (legacy) donations.
1.4 The system shall automatically bill donors on legacy donation plans.
1.5 The system shall accept payment from credit card and e-check accounts.

2. Non-Monetary Donations

2.1 The system shall accept non-monetary donations.
2.2 The system shall keep a running list of needed items for G.I.K. donations.

3. Donor Information

3.1 The system shall accept new donors and allow the user to edit current donor information (address, phone, etc.) in a database.
3.2 The system shall be integrated with the NCOA system (National Change of Address)
3.2 The system shall update donor accounts when new donations are received.

4. Campaigns

4.1 The system shall manage marketing & newsletter campaigns.

5. Wish List

5.1 The system shall maintain a running list of needed items.

6. Volunteer Information

6.1 The system shall track the information and activities of local volunteers.
6.2 The system shall track the information and activities of immersion trip volunteers.

4.2 Non-Functional Requirements

1. Operational Requirements

1.1 The system shall use only one contact table that will be shared by all programs and databases.
1.2 The system shall be compatible with software that is currently in use (i.e. Microsoft Excel)
1.3 The system shall operate in either Windows or Mac environments.

2. Performance Requirements

2.1 The system shall be operational 95% of the time.
2.2 The system shall retrieve donor records with an average response time of 5 seconds or less.
2.3 The system shall lookup values and return the results with an average response time of 5 seconds or less.

3. Hardware Requirements

3.1 The system shall use personal Computers (Mac or PC) manufactured after 2005 with a minimum of 1Gb RAM.
3.2 The system shall have broadband access, routers, modem, and service.
3.3 The system shall be equipped with either wi-fi or Ethernet cards to interact with the donor database.
3.4 The system shall use small I.D. scanners (one per donation site)
3.5 The system shall use UPC scanners (minimum one per donation site)
3.6 The system shall have Ink-Jet or Laser printers (a minimum of one per site)

4. Software Requirements

4.1 The system shall use an Internet Browser with 128-bit encryption. Firefox is recommended.
4.2 The system shall run on Windows XP SP3/Windows Vista/Windows 7 or Mac OSX

5. Processes to Be Automated

1. Legacy Donation transactions and scheduling shall be automated
2. The G.I.K. valuation system shall be automated.
3. Receipt generation shall be automated
4. Donor records shall be updated automatically with new donations.
5. The G.I.K. "curb-side" recording process shall be automated.

Requirements Tracing Use Case ID	1	2	3	4	5	6	7	8	
Requirement									
1.1		X							
1.2		X							
1.3		X							
1.4		X							
1.5		X							
2.1			X						
2.2			X				X		
3.1				X					
3.2				X	X			X	X
3.3				X					
4.1				X	X				
5.1						X			
6.1								X	
6.2									X

1. Monetary Donations

1.1 The system shall accept monetary donations.

1.2 The system shall provide documentation of transactions automatically.

1.3 The system shall accept and manage multi-year (legacy) donations.

1.4 The system shall automatically bill donors on legacy donation plans.

1.5 The system shall accept payment from credit card and e-check accounts.

2. Non-Monetary Donations

2.1 The system shall accept non-monetary donations.

2.2 The system shall keep a running list of needed items for G.I.K. donations.

3. Donor Information

3.1 The system shall accept new donors and allow the user to edit current donor information (address, phone, etc)

3.2 The system shall be integrated with the NCOA system (National Change of Address)

3.2 The system shall update donor accounts when new donations are received.

4. Campaigns

4.1 The system shall manage marketing & newsletter campaigns.

5. Wish List

5.1 The system shall maintain a running list of needed items.

6. Volunteer Information

6.1 The system shall track the information and activities of local volunteers.

6.2 The system shall track the information and activities of immersion trip volunteers.

Use Case Descriptions

Accept Monetary Donation

The Accept Monetary Donation use case is used is the description that an HHM Associate has with the system when a donation is made. The Associate must enter the type of donation, One Time or Legacy as well as entering the payment information and providing the donor with a receipt for the transaction.

Use Case: Accept Monetary Donation	**ID:** 1
Risk level: High	**Use Case Type:** Essential, Detail
Brief Description: A contact has made a monetary donation.	
Primary Actors: HHM Associate	
Secondary Actors: None	
Pre-conditions: A monetary donation has been made to HHM.	
Main Flow: 1. Extend Manage Contact 2. The HHM Associate selects the contact to link the donation to. 3. The system prompts the HHM Associate to enter the applicable donation information. 4. The HHM Associate enters the information, for example the donation amount, type (legacy or one time), and payment method and saves the donation. 5. The system updates the contact history and prints the donation receipt. 6. Optional Flow: 7. The HHM Associate prompts the system to create a Deposit Slip and or Daily Donation Report. 8. The system prints the reports	
Post conditions: The donation has been recorded, a receipt has been printed,	
Alternative Flows: None	

Accept Gift in Kind Donation

The Accept Gift in Kind Donation use case describes the interaction between the system and HHM Associate when a contact makes a Gift In Kind Donation. The system will specifically add items to the Gift in Kind inventory, appraise in-kind donations, assign the donation to an existing contact, and print a receipt. The quantity and description are added to the in-kind inventory and the wish list is update to reflect the donation automatically. The in-kind gift donation form used with a UPC scanner or entered manually. If a contact is not in the system prior to the in-kind donation, a contact must be created first.

Use Case: Accept In Kind Donation	**ID:** 2
Risk level: High	**Use Case Type:** Essential, Detail
Brief Description: The system provides donor receipt, stores gift value and updates in-kind inventory.	
Primary Actors: HHM Associate	
Secondary Actors: None	
Pre-conditions: A contact has presented HHM With an In Kind Donation	
Main Flow: 1. Include (Manage Contact) 2. The HHM Associate selects the contact to link the donation to. 3. The system prompts the HHM Associate to enter the applicable donation information. 4. The HHM Associate enters the UPC code by typing the code in to a text box or scanning the bar code with a UPC scanner. 5. If no barcode or UPC exists, the HHM Associate enters the description in to the system. 6. The HHM Associate enters the items condition and the webpages to reference the items value. 7. The system returns the selected providing the HHM Associate enough information to make a Good Faith Estimate. 8. The HHM Associate enters the website used and the amount for the Good Faith Estimate. 9. The System provides the HHM Associate the option to remove the donated item from the wish list. 10. The HHM Associate saves the donation. 11. The system prints the receipt.	
Post conditions: The system has recorded the In Kind Donation in In Kind Inventory and printed a receipt for the donor (contact).	

Manage Contacts

The Manage Contact Use-Case is the interaction an HHM Associate has with the system when updating an existing donor or creating new donor. Entering a new donor and editing an existing donor are similar. The difference between entering a new donor and editing an existing donor is that editing an existing donor attribute begins with pre-filled (by system) attribute details for the donor selected for editing. When a new donor is created, the attribute fields contain a value equal to null. If the system contains duplicate entries for existing donors, at the HHM Associates request the system shall merge the two donors to create a single donor. When merging donors the HHM Associate will have the ability to select which attribute entries will continue to exist in the new merged donor. If Donor A and Donor B is the same person, the HHM Associate will select either Donor A or Donor B to be the Primary Donor and thus the Secondary Donor. By default, the attribute fields from the primary donor will have precedence in the merged donor unless the HHM Associate specifies attribute fields from the Secondary Donor to overwrite fields in the Primary donor. All recorded activity and existing historical records and from the Secondary Donor shall merge with the recorded activity and historical recodes in the Primary Donor.

Use Case: Manage Contacts	**ID:** 3
Importance level: High	**Use Case Type:** Essential, Detail
Primary Actors: HHM Associate	
Brief Description: Search for a specific contact.	
Trigger: An HHM Associate needs to create, edit, or merge donors.	
Secondary Actors: None	
Pre-conditions: HHM has interacted with a contact and needs to locate the contacts information within the system.	
Main Flow: 1. The HHM Associate prompts the system to display the Search Contact interface 2. The system displays the Search Contact interface. The HHM Associate enters an attribute of the contact in the search bar and clicks the Search Contacts button. 3a. If the system displays the contacts with matching attributes 3b. If the system displays a message that no matching contacts exist. 4a. The HHM Associate selects the contact for editing OR Selects two contacts to merge. 4b. The HHM Associates chooses to create a new contact. 5. The system displays the Manage Contact form with the selected contacts attributes pre filled OR the system displays the Manage Contact form with blank attribute fields. 6. The HHM Associate edits the necessary attributes and clicks the save button. 7. The system saves the updated attribute with a unique identifier.	
Post conditions: The system has saved the information entered buy the HHM Associate.	

Create Mailing List

The create mailing list use case details the steps a HHM Associate will take to select the appropriate criteria to return a list of addresses that match the organization's needs. The "Ad-Hoc report manager" set up to perform this performs this function. The report manager will be adjusted for specific tasks such as creating a mailing list or searching donors for any other reason. In the example that follows (prototype) we have set the manager to return the donor state as KY, the donation date as before the first of 2009, OR total donations at less than $500 over the life of the donor's account. This will return a list of donors that fit that criteria and mailings can be targeted at them. Below is the use case description, the prototype follows on the next page.

Use Case: Create Mailing List	**ID:** 4
Importance Level: Low	**Use Case Type:** Essential, Detail
Brief Description: Create Mailing List allows an HHM Associate to query the contact database to return contacts meeting specific criteria.	
Primary Actors: HHM Associate	
Secondary Actors: None	
Pre-conditions: HHM needs a list of contacts meeting certain criteria.	
Main Flow: 1. The HHM Associate prompts the system to display the Mailing List interface 2. The system prompts the HHM Associate to enter the criteria for querying the contact database. 3. The HHM Associate enter the attributes. 4. The database returns the results. 5. The HHM Associate prompts the system to save the report and export the list to Microsoft Excel.	
Post conditions: The campaign has been created and saved	

Manage Wish List

The Manage Wish List use case describes the interaction the HHM Associate has with the system when creating or changing Wish Lists. Wish Lists are composed of items HHM needs to fulfill their mission. The list contains objects such as personal items, food, or construction tools. To influence the Gift In Kind donations HHM publishes the list on their website. In addition, the new system will be able to update donors when new items are added or a new wish list is created. The notification can be done via text message, post mail, or email.

Use Case: Manage Wish List	**ID:** 5
Risk level: Low	**Use Case Type:** Essential, Detail
Brief Description: Create, edit, and delete Wish Lists.	
Primary Actors: HHM Associate	
Secondary Actors: None	
Pre-conditions:	
Main Flow: 1. The HHM Associate prompts the system to display the wish list interface. 2. The system displays the interface. 3. The HHM Associate chooses the wish list to modify. 4. If the desired wish list does not exist, the HHM Associate creates a new wish list. 5. The system displays the items in the selected wish list. 6. The HHM Associate selects to create new items. 7. The HHM Associate selects an item from the wish list to modify. 8. The system displays the attributes in editable textboxes. 9. The HHM Associate edits the attributes. 10. The system asks the HHM Associate where to publish the changes. 11. The HHM Associate selects the following option(s) Publish to Website, Send Update to contacts via E- mail, Post Mail, and Text Message. The HHM Associate also selects the Mailing List to update.	
Post conditions: The wish list has been updated on the web sites and contacts have been notified.	
Alternative Flows: None	

Manage Gift-In-Kind Inventory

Use-Case Manage GIK Inventory is the interaction an HHM Associate has with the system when making modifications to the recorded inventory attribute values. Inventory exists in two forms physical and recorded. If discrepancy between a recorded inventory items attribute value and the actual physical quantity available in HHM's inventory it is necessary to modify the recorded inventory attributes accordingly. The recorded inventory attributes include a unique identifier, item description, quantity available and the region to which the items are allocated to. A recorded inventory serves several purposes. One, items in the physical inventory may be assigned to (or claimed by) a particular region. It is assumed that the charities must be creative when using resources to help those in need and viewing the recorded inventory with the ability to claim certain items will inspire creative ways HHM Associates can allocate limited resources. Two, a recorded inventory creates an inventory control to protect against inventory shrinkage (theft). Three, publishing the recorded inventory allows all HHM Associates to be aware of the items allocation status thus reducing the number of communication channels.

Use Case: Manage In Kind Inventory	**ID:**6
Importance level: High	**Use Case Type:** Essential, Detail
Primary Actors: HHM Associate	
Brief Description: Donor management includes adding items to gift in kind inventory and assigning those items to a region.	
Trigger: An HHM Associate needs update the quantity, description, or assigned region of an item(s).	
Secondary Actors: None	
Pre-conditions: HHM has received an In Kind Donation.	
Main Flow: 1. The HHM Associate opens the Manage In Kind Inventory form. 2. The system displays all inventory items including the description, quantity and the region the items are assigned to. 3. The HHM Associate confirms that the physical quantity does not match the reported quantity and selects which items to edit. 5. The system displays the selected items and editable textboxes in a new form for the HHM Associate to update. 6. The HHM Associate updates necessary items and includes and clicks the save button. 7. The system updates the inventory and returns the HHM Associate to the Manage In Kind Inventory interface. **Alternative Flow** 2a. The HHM Associate enters a description in the search box. 3a. The system searches the inventory for a matching item. 4a. The HHM Associate edits the item attributes and clicks save. 5a. The system saves the updated item.	
Post conditions: The system has updated the In Kind Inventory to match the information entered buy the HHM Associate.	

Track Local Volunteer

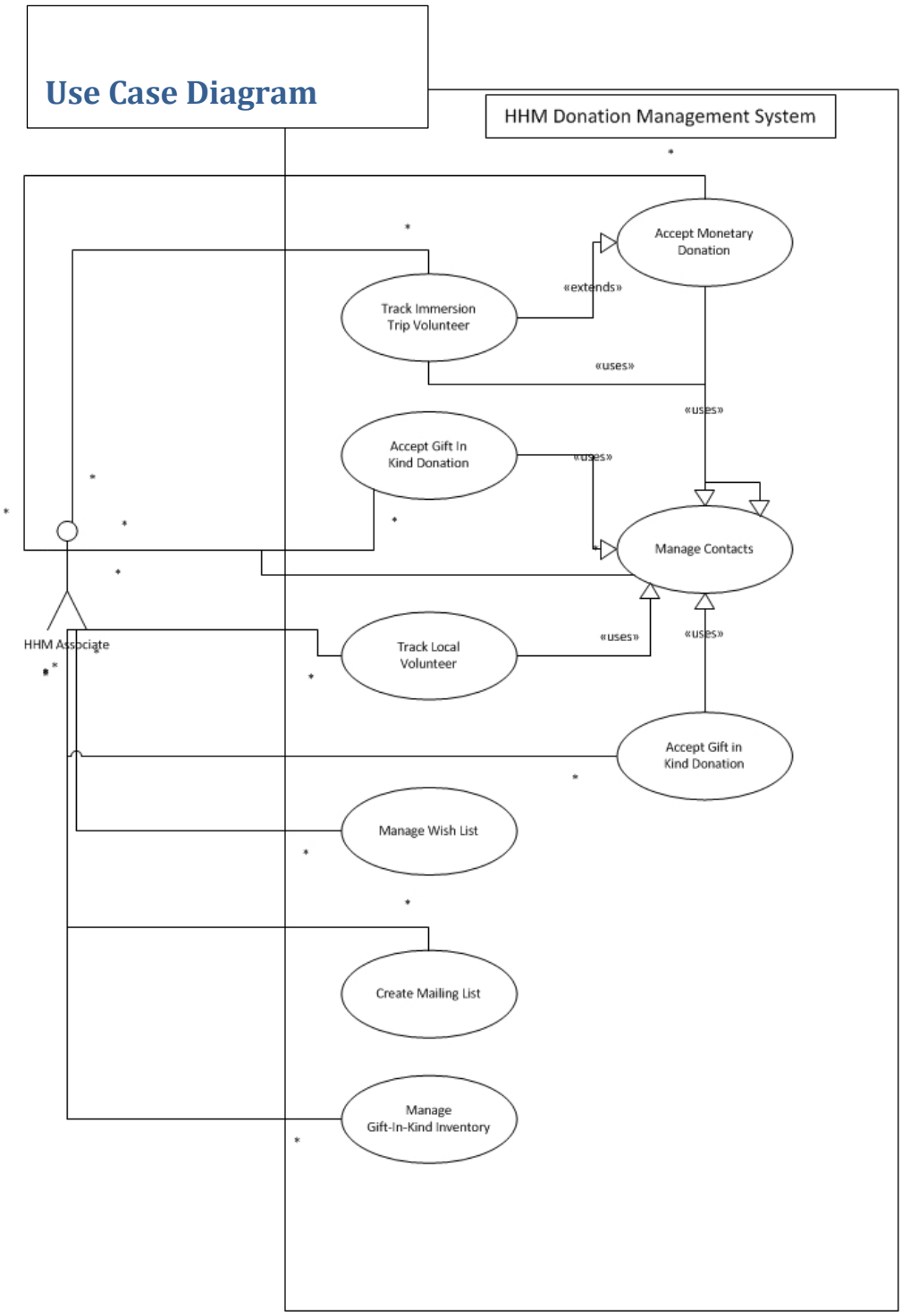
This use case is for tracking and managing local volunteers. When the HHM Associate selects create/manage volunteer the system prompts the HHM Associate to select which type of volunteer to be updated or created. The options are "local volunteer" or "Immersion Trip Volunteer." After selecting the appropriate volunteer, the HHM Associate can search for an existing volunteer or add a new one. The HHM Associate will then input the data they want recorded. The information that can be recorded is found in the track local volunteer prototype. The HHM Associate then ends the session and returns to the main menu.

Use Case: Track Local Volunteer	ID:2
Primary Actors: HHM Associate	**Use Case Type:** Essential, Detail
Brief Description: The system will be able to update / create new volunteer in the system.	
Trigger: A need to access or update local volunteer record is identified.	
Secondary Actors: None	
Pre-conditions: The volunteer must exist as a contact in the system.	
Main Flow: 1. Extend Manage Contact 2. The system displays the contacts with matching attributes and a list of the dates and time the contact previously volunteered. 3. The HHM Associate selects the contact to view or create the local volunteer attributes. 4. The system displays the Track Local Volunteer form with previously saved volunteer attributes or blank attribute fields if the contact is a new local volunteer 5. The HHM Associate edits the necessary attributes for example the time and or the day the contact is available to volunteer and the location of where the contact prefers to volunteer. 6. The system saves the updated attributes.	
Post conditions: The system has saved the information entered buy the HHM Associate.	

Track Immersion Trip Volunteer

This use case is for tracking and managing local volunteers. When the HHM Associate selects create/manage volunteer the system prompts the HHM Associate to select which type of volunteer to be updated or created. The options are "local volunteer" or "Immersion Trip Volunteer." After selecting the appropriate volunteer, the HHM Associate can search for an existing volunteer or add a new one. The HHM Associate will then input the data they want recorded. The information that can be recorded is found in the track immersion trip volunteer prototype. The HHM Associate then ends the session and returns to the main menu.

Use Case: Track Immersion Trip Volunteer	**ID:**8
Importance level: Medium-High	**Use Case Type:** Essential, Detail
Primary Actors: HHM Associate	
Brief Description: The system will be able to update / create new Immersion Trip Volunteer in the system.	
Trigger: A contact has informed HHM that they would like to participate on an immersion trip.	
Pre-conditions: The volunteer must exist as a contact in the system.	
Main Flow: 1. Extend Manage Contact 2. The system displays the Track Immersion Trip Volunteer and the contact is now linked to the information updated by the HHM Associate. 3. The HHM Associate enters the contacts passport country, passport number, emergency contact information and the immersion trip that the contact will participating in. 4. The system prompts the HHM Associate to enter the information about the required donation for the immersion trip. **Extension Point:** Accept Monetary Donation 5. The system saves the updated attributes.	
Post conditions: The system has saved the information entered buy the HHM Associate.	



Use Case Narrative

The use case diagram is an illustration of how each individual business process is related within the system. The actor depicts a generic HHM Associate. The border around the use cases is the system. The actor is outside of the system to provide inputs and receive output.

Create Mailing List

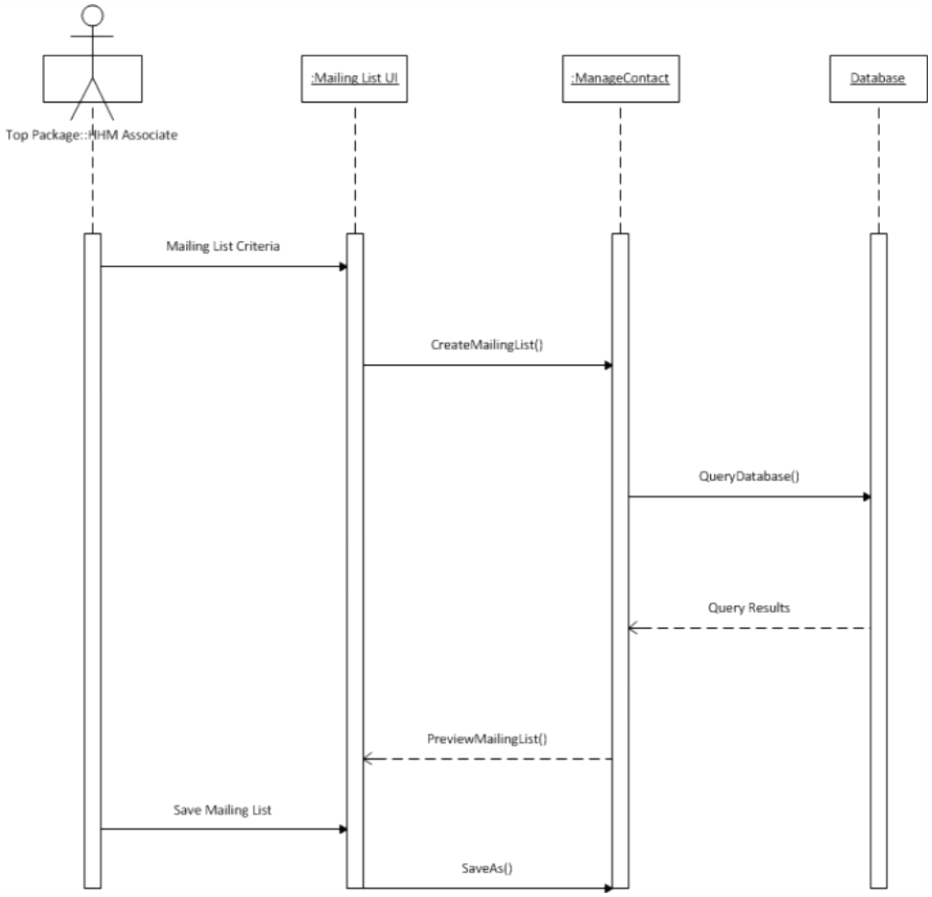
Create Mailing List

1. The HHM Associate prompts the system to display the Mailing List interface

2. The system prompts the HHM Associate to enter the criteria for querying the contact database.
3. The HHM Associate enter the attributes.

4. The database returns the results.

5. The HHM Associate prompts the system to save the report and export the list to Microsoft Excel.



Track Local Volunteer

1. Extend Manage Contact

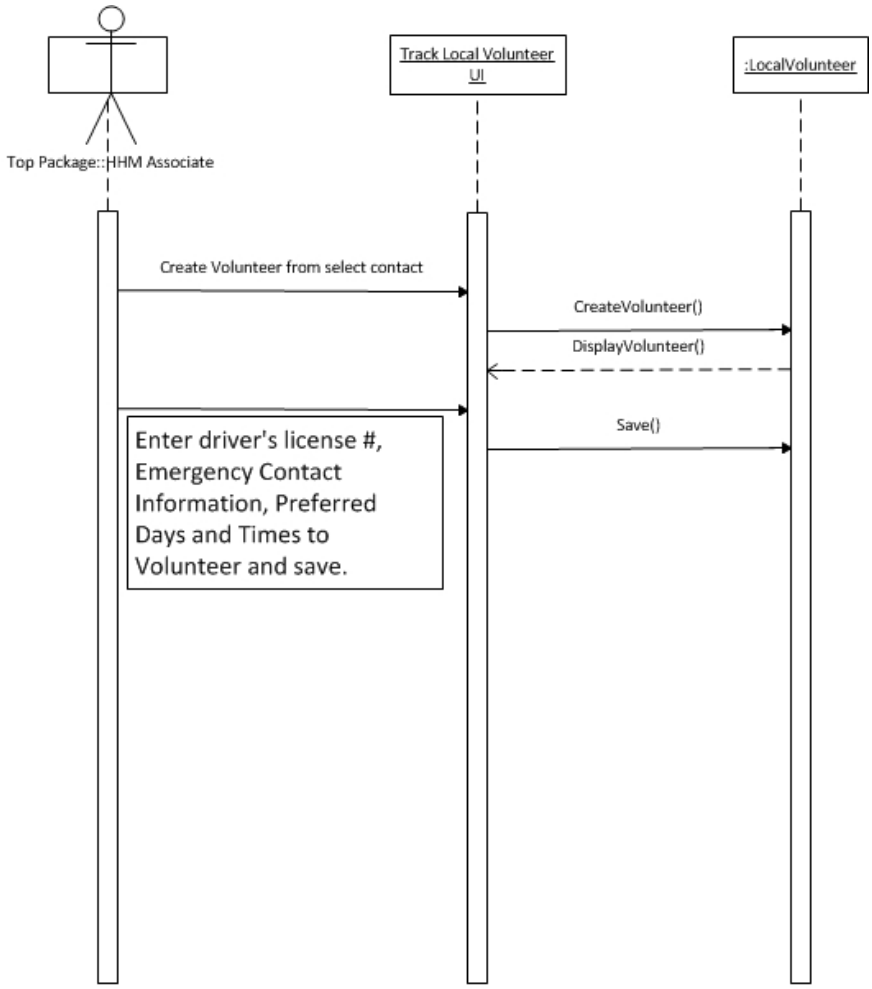
2. The system displays the contacts with matching attributes and a list of the dates and time the contact previously volunteered.

3. The HHM Associate selects the contact to view or create the local volunteer attributes.

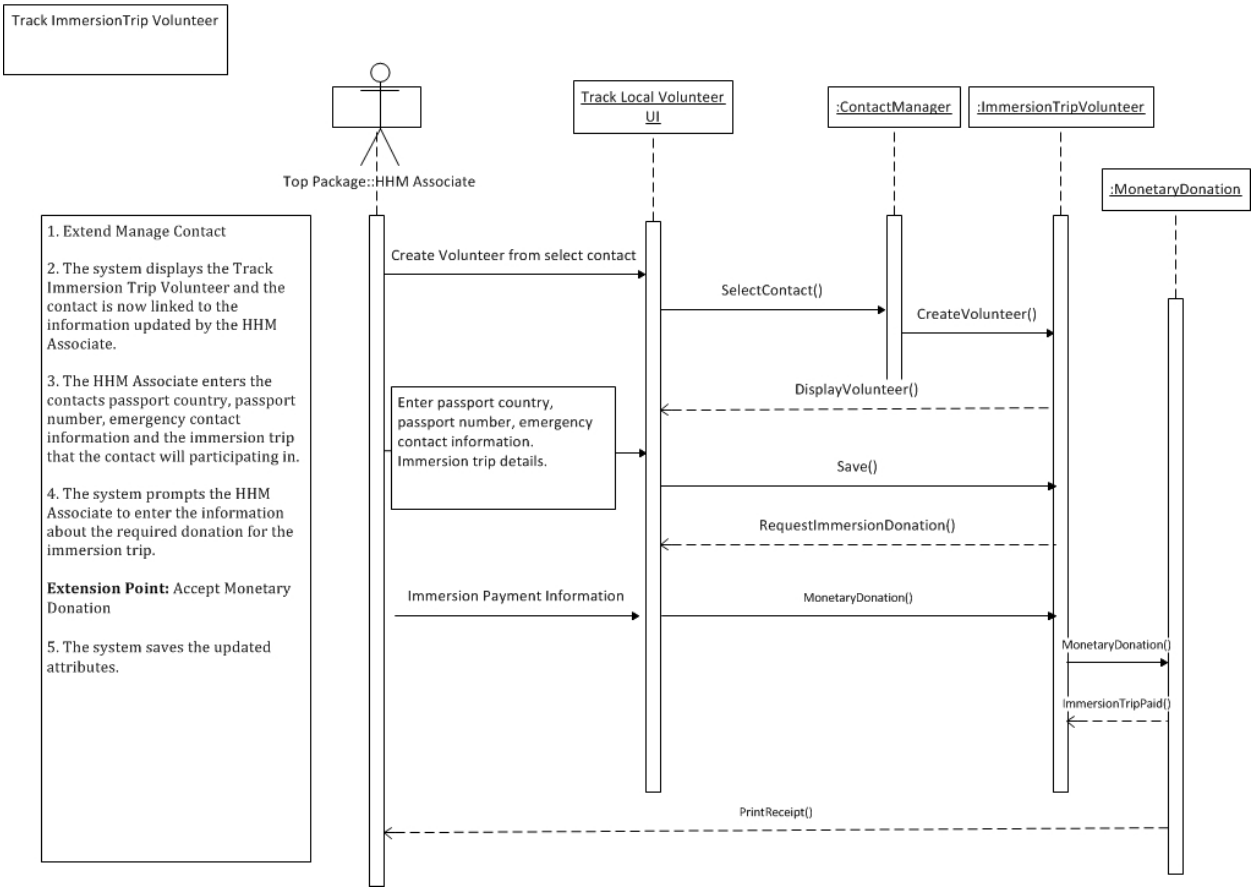
4. The system displays the Track Local Volunteer form with previously saved volunteer attributes or blank attribute fields if the contact is a new local volunteer

5. The HHM Associate edits the necessary attributes for example the time and or the day the contact is available to volunteer and the location of where the contact prefers to volunteer.

6. The system saves the updated attributes.



Track Immersion Trip Volunteer


Track ImmersionTrip Volunteer
Top Package::HHM Associate
Track Local Volunteer UI
:ContactManager
:ImmersionTripVolunteer
:MonetaryDonation
1. Extend Manage Contact
2. The system displays the Track Immersion Trip Volunteer and the contact is now linked to the information updated by the HHM Associate.
3. The HHM Associate enters the contacts passport country, passport number, emergency contact information and the immersion trip that the contact will participating in.
4. The system prompts the HHM Associate to enter the information about the required donation for the immersion trip.
Extension Point: Accept Monetary Donation
5. The system saves the updated attributes.
Create Volunteer from select contact
SelectContact()
CreateVolunteer()
DisplayVolunteer()
Enter passport country, passport number, emergency contact information. Immersion trip details.
Save()
RequestImmersionDonation()
Immersion Payment Information
MonetaryDonation()
MonetaryDonation()
ImmersionTripPaid()
PrintReceipt()

Manage Gift In Kind

Manage GIK Inventory
Sequence Diagram

Main Flow:
1. The HHM Associate opens the Manage In Kind Inventory form.

2. The system displays all inventory items including the description, quantity and the region the items are assigned to.

3. The HHM Associate confirms that the physical quantity does not match the reported quantity and selects which items to edit.

5. The system displays the selected items and editable textboxes in a new form for the HHM Associate to update.

6. The HHM Associate updates necessary items and includes and clicks the save button.

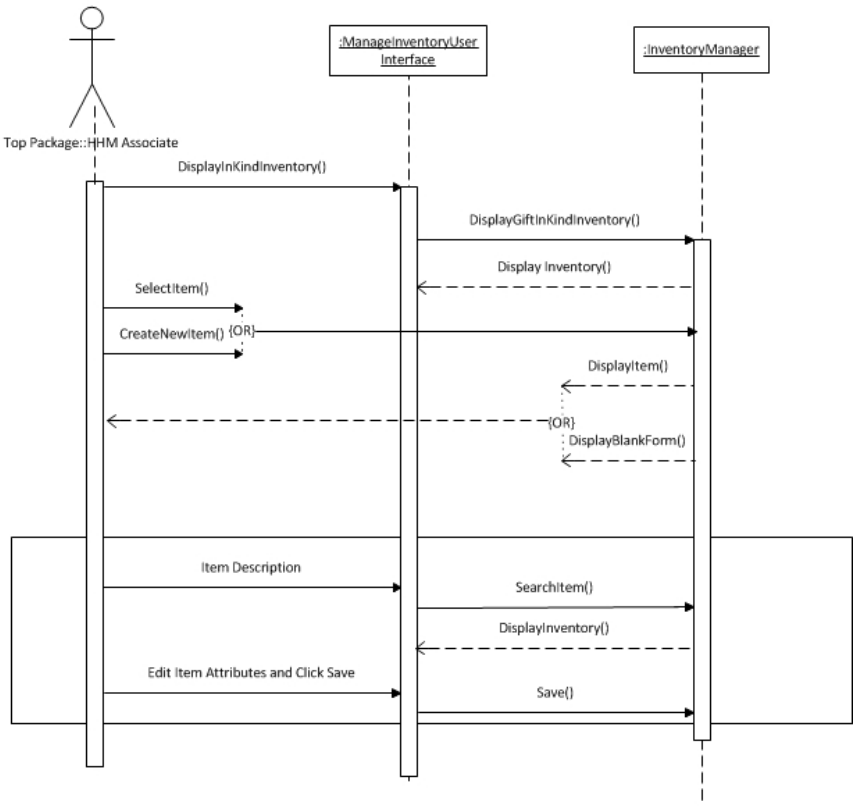
7. The system updates the inventory and returns the HHM Associate to the Manage In Kind Inventory interface.

Alternative Flow
2a. The HHM Associate enters a description in the search box.

3a. The system searches the inventory for a matching item.

4a. The HHM Associate edits the item attributes and clicks save.

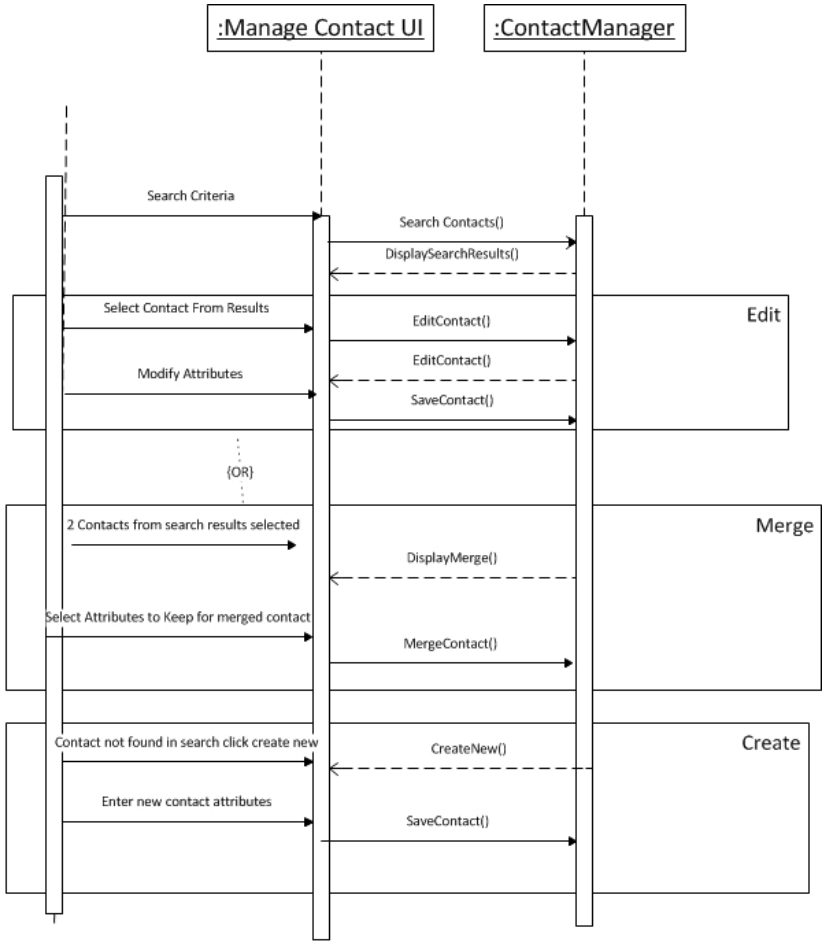
5a. The system saves the updated item.



Manage Contact

Manage Contact
Sequence Diagram

1. The HHM Associate prompts the system to display the Search Contact interface
2. The system displays the Search Contact interface.
3. The HHM Associate enters an attribute of the contact in the search bar and clicks the Search Contacts button.
4a. If the system displays the contacts with matching attributes
    4b. If the system displays a message that no matching contacts exist.
5a. The HHM Associate selects the contact for editing **OR** Selects two contacts to merge.
    5b. The HHM Associates chooses to create a new contact.
6. The system displays the Manage Contact form with the selected contacts attributes pre filled OR the system displays the Manage Contact form with blank attribute fields.
7. The HHM Associate edits the necessary attributes and clicks the save button.
8. The system saves the updated attribute with a unique identifier.



Manage Wish List

1. The HHM Associate prompts the system to display the wish list interface.

2. The system displays the interface.

3. The HHM Associate chooses the wish list to modify.

4. If the desired wish list does not exist, the HHM Associate creates a new wish list.

5. The system displays the items in the selected wish list.

6. The HHM Associate selects to create new items.

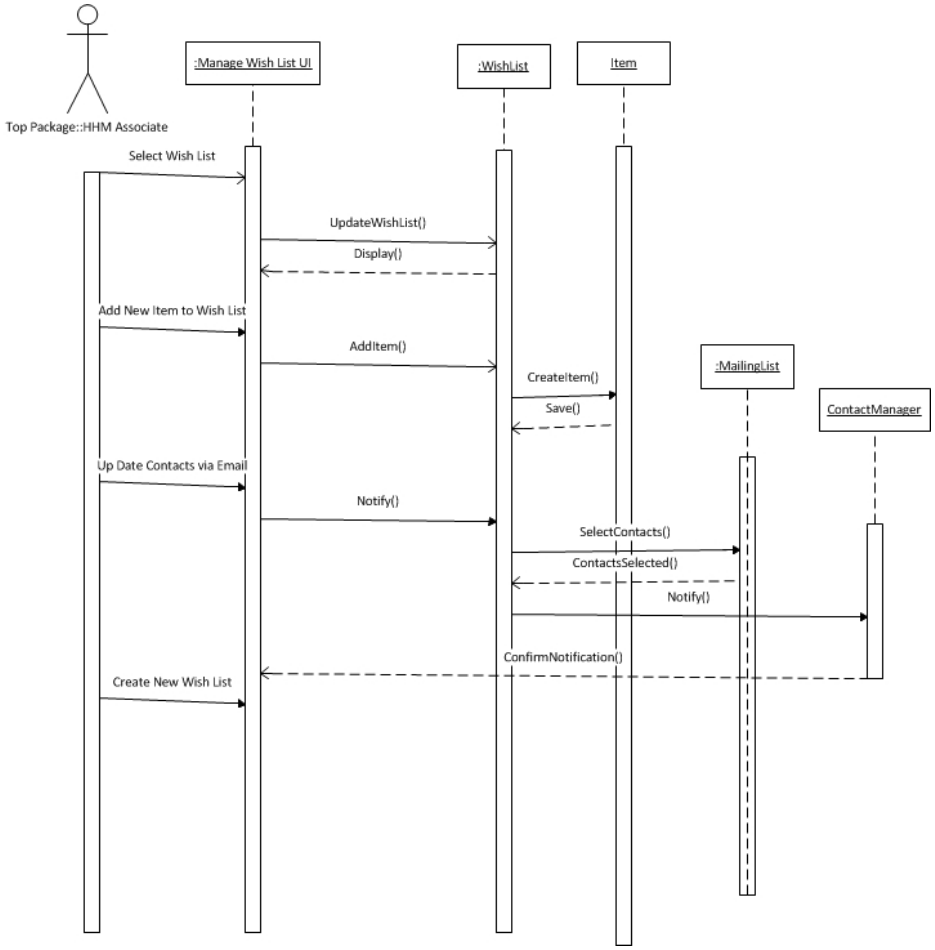
7. The HHM Associate selects an item from the wish list to modify.

8. The system displays the attributes in editable textboxes.

9. The HHM Associate edits the attributes.

10. The system asks the HHM Associate where to publish the changes.

11. The HHM Associate selects the following option(s) Publish to Website, Send Update to contacts via E- mail, Post Mail, and Text Message. The HHM Associate also selects the Mailing List to update.



Accept In Kind Donation

Accept Gift In Kind Donation

1. Include (Manage Contact)

2. The HHM Associate selects the contact to link the donation to.

3. The system prompts the HHM Associate to enter the applicable donation information.

4. The HHM Associate enters the UPC code by typing the code in to a text box or scanning the bar code with a UPC scanner.

5. If no barcode or UPC exists, the HHM Associate enters the description in to the system.

6. The HHM Associate enters the items condition and the webpages to reference the items value.

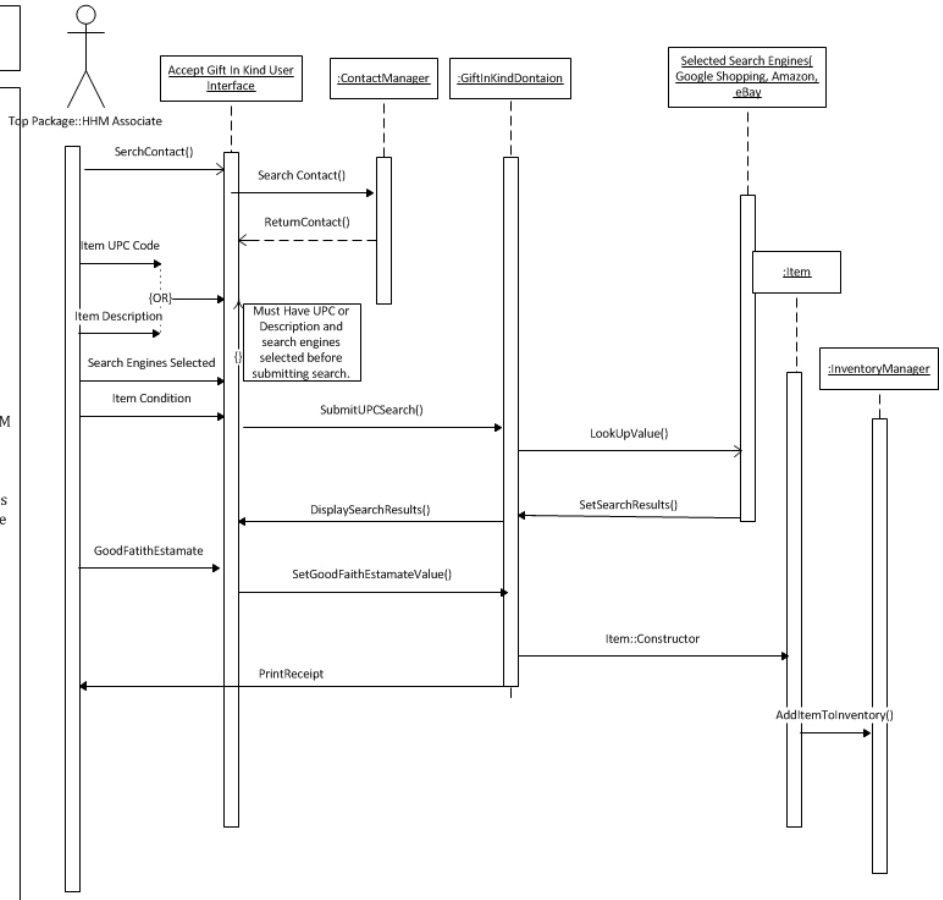
7. The system returns the selected providing the HHM Associate enough information to make a Good Faith Estimate.

8. The HHM Associate enters the website used and the amount for the Good Faith Estimate.

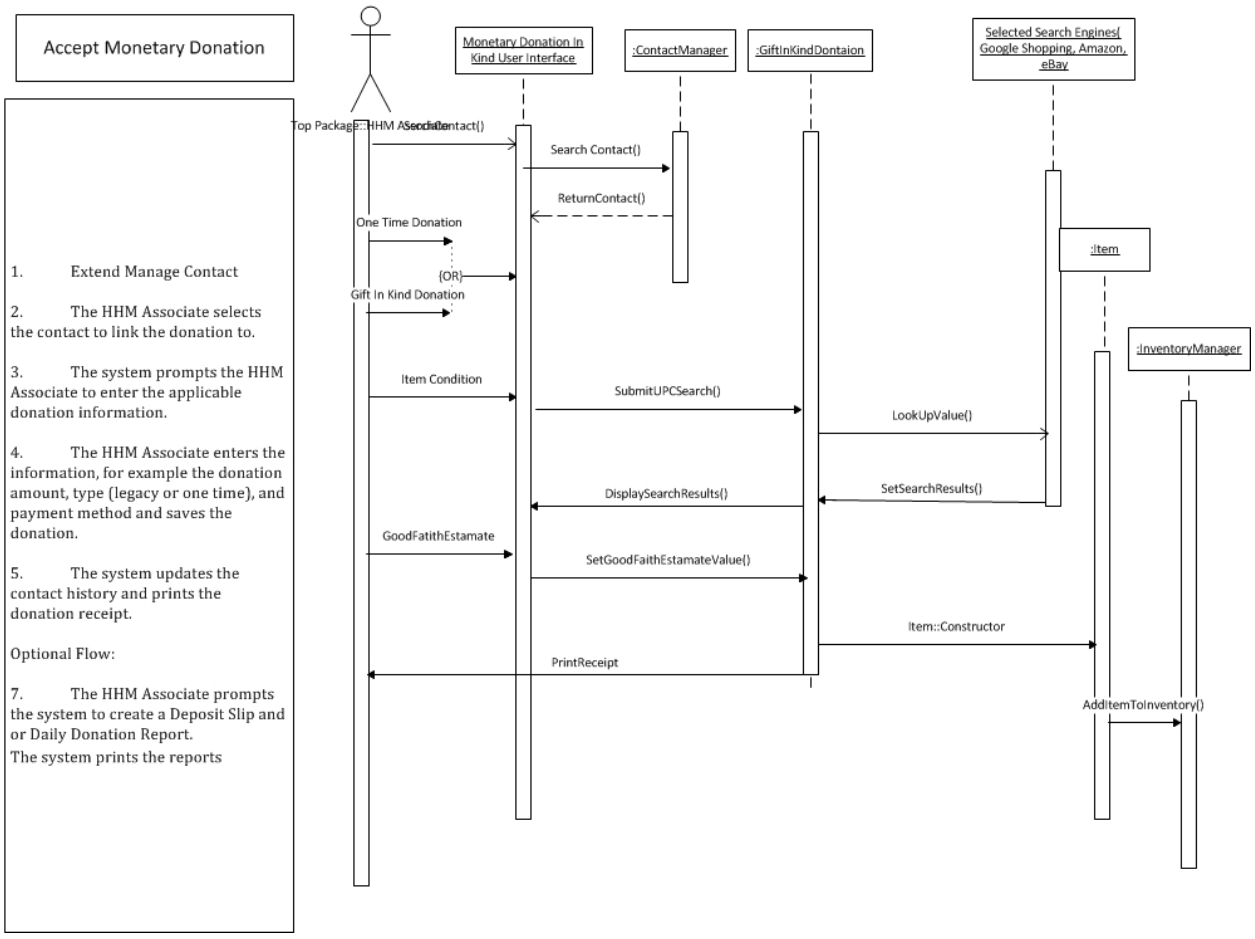
9. The System provides the HHM Associate the option to remove the donated item from the wish list.

10. The HHM Associate saves the donation.

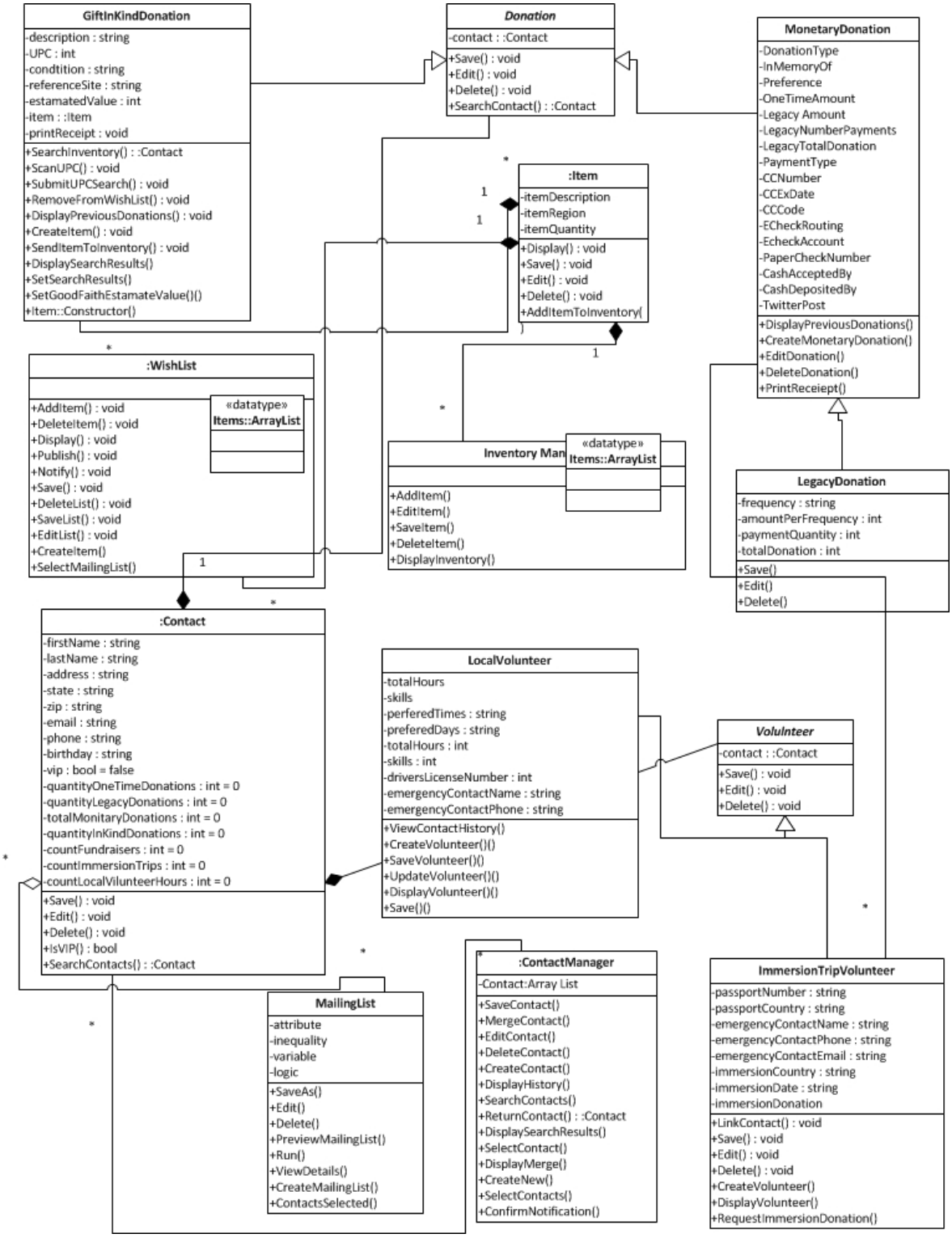
11. The system prints the receipt.



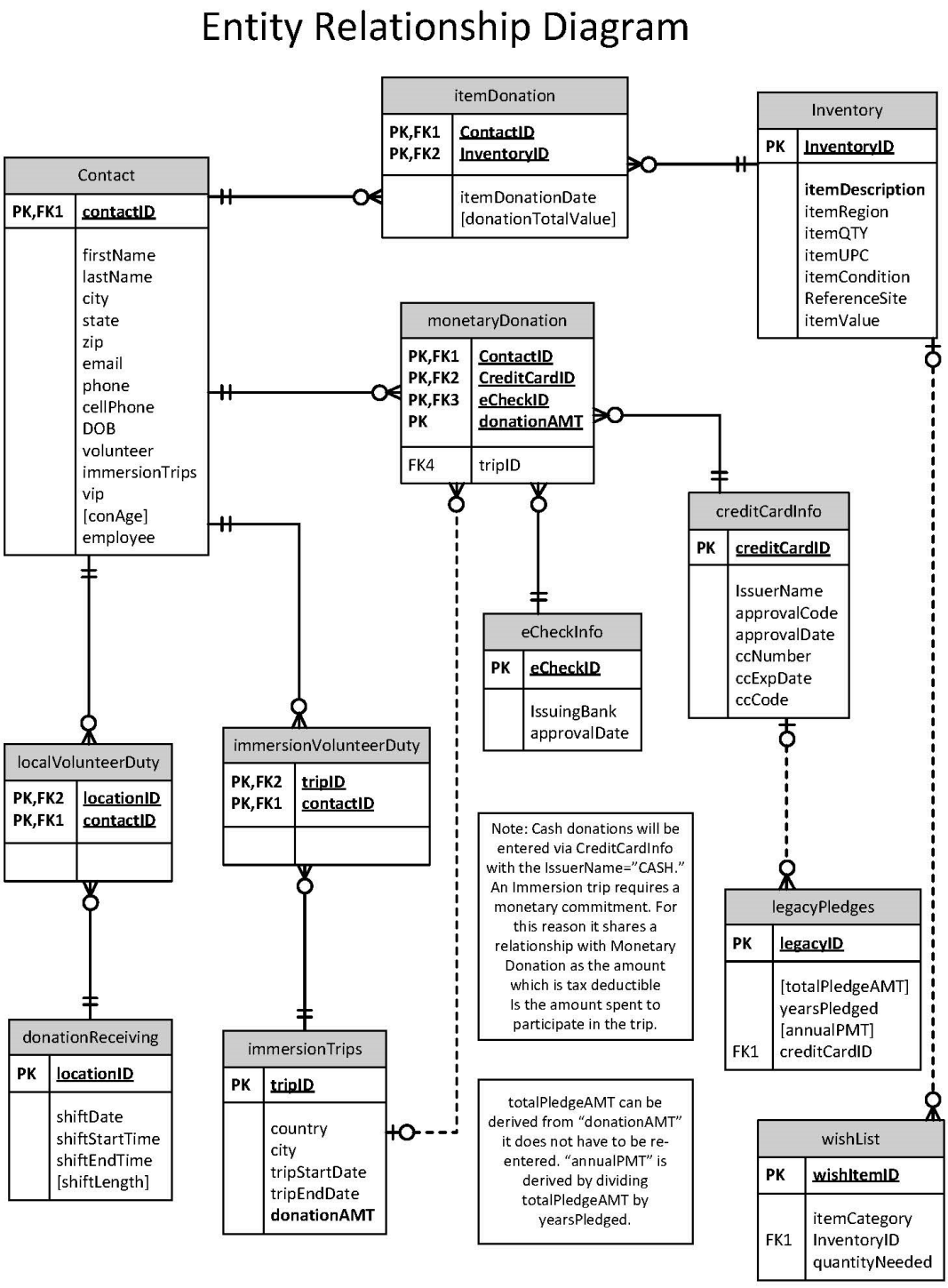
Accept Monetary Donation


Accept Monetary Donation
1. Extend Manage Contact
2. The HHM Associate selects the contact to link the donation to.
3. The system prompts the HHM Associate to enter the applicable donation information.
4. The HHM Associate enters the information, for example the donation amount, type (legacy or one time), and payment method and saves the donation.
5. The system updates the contact history and prints the donation receipt.
Optional Flow:
7. The HHM Associate prompts the system to create a Deposit Slip and or Daily Donation Report.
The system prints the reports
Monetary Donation In Kind User Interface
:ContactManager
:GiftInKindDontaion
Selected Search Engines(Google Shopping, Amazon, eBay
:Item
:InventoryManager
Search Contact()
ReturnContact()
One Time Donation
{OR}
Gift In Kind Donation
Item Condition
SubmitUPCSearch()
LookUpValue()
SetSearchResults()
DisplaySearchResults()
GoodFatithEstamate
SetGoodFaithEstamateValue()
Item::Constructor
PrintReceipt
AddItemToInventory()

Class Diagram



Database Design



Data Definitions

This table is laid out in alphabetical order. Refer to the Entity Relationship Diagram to view the placement of these entities and attributes within the database system.

name	type	name	type
annualPMT:	currency	itemQTY:	integer
approvalCode:	varchar(10)	itemRegion:	varchar(10)
approvalDate:	datetime	itemUPC:	integer
approvalDate:	datetime	itemValue:	currency
ccCode:	integer	lastName:	varchar(10)
ccExpDate:	datetime	legacyID:	integer
ccNumber:	integer	locationID:	char(10)
cellPhone:	integer	phone:	integer
city:	char(10)	quantityNeeded:	integer
city:	varchar(15)	referenceSite:	varchar(10)
conAge:	integer	shiftDate:	datetime
contactID:	integer	shiftEndTime	datetime
creditCardID:	integer	shiftLength:	decimal(2,2)
DOB:	datetime	shiftstartTime:	datetime
donationAMT:	currency	state:	varchar(10)
email:	varchar(20)	totalDonationValue:	currency
employee:	bit	totalPledgeAMT:	currency
firstName:	varchar(10)	tripEndDate:	datetime
immersionTrips:	bit	tripID:	char(10)
inventoryID	integer	tripLength:	decimal(2,2)
issuerName:	varchar(10)	tripStartDate:	datetime
issuingBank:	char(10)	vip:	bit
itemCategory:	char(10)	volunteer:	bit
itemCondition:	varchar(10)	yearsPledged:	integer
itemDescription:	varchar(30)	zip:	integer
itemDonationDate:	datetime		

Screen Layouts

Main Menu

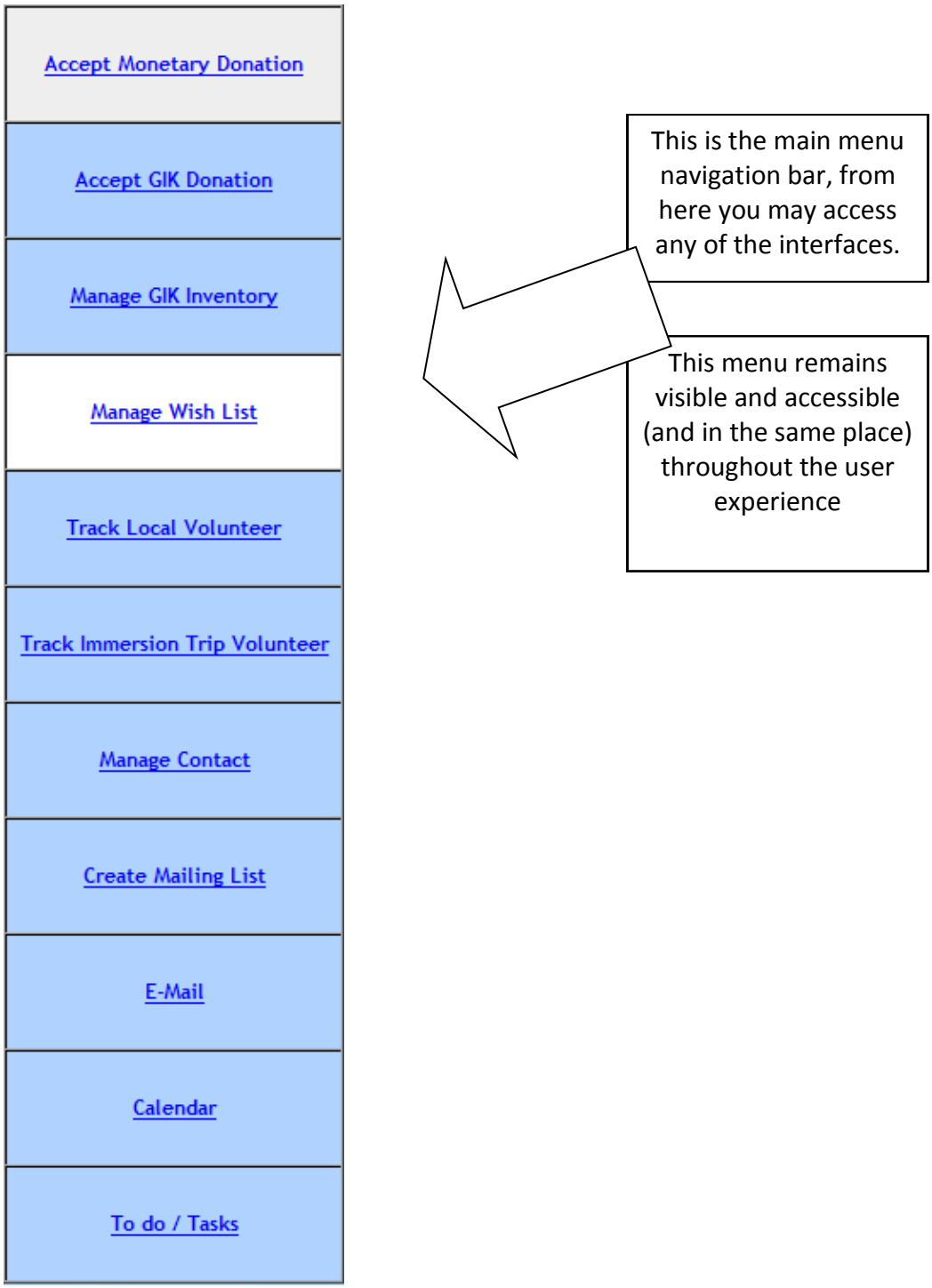
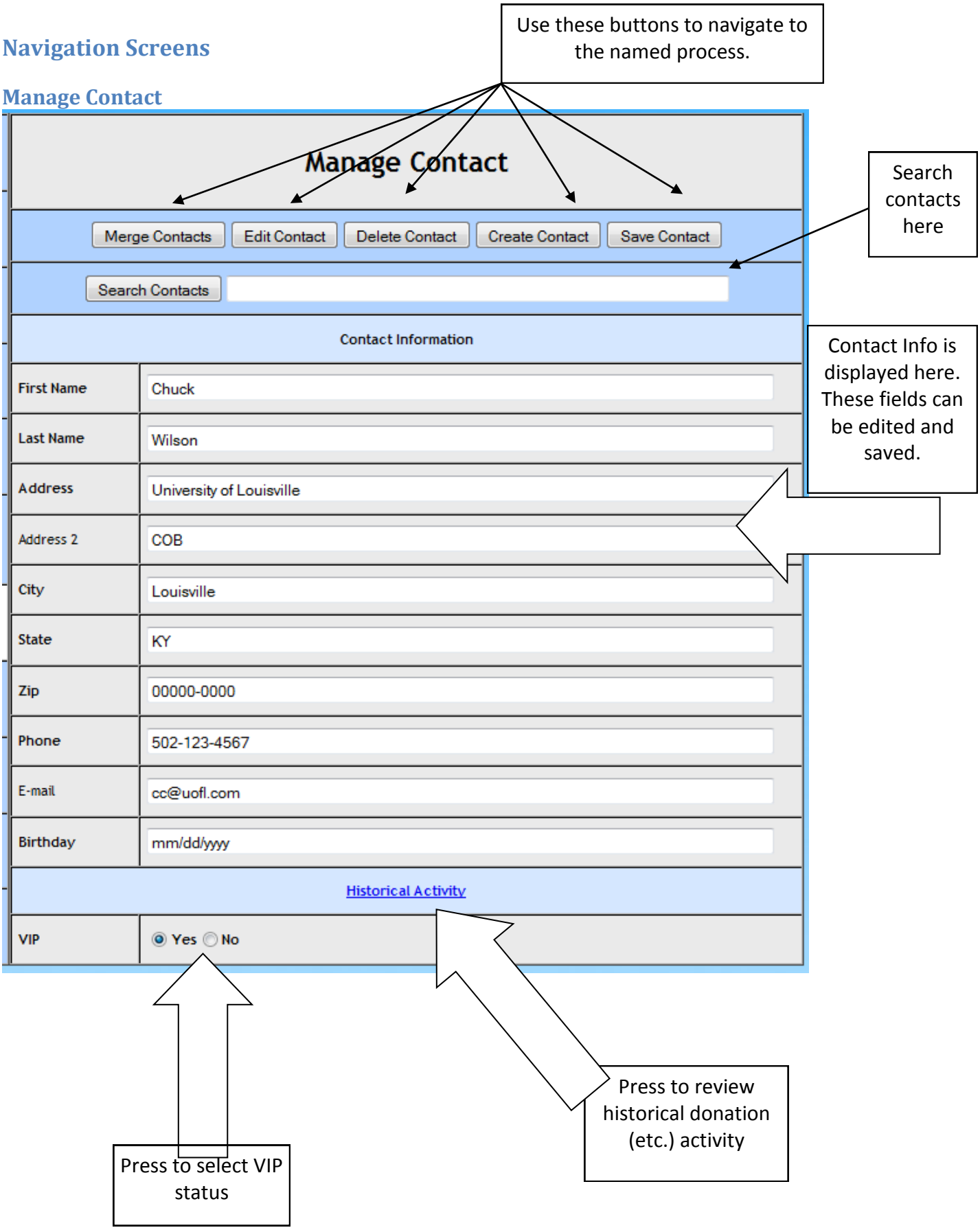

Accept Monetary Donation
Accept GIK Donation
Manage GIK Inventory
Manage Wish List
Track Local Volunteer
Track Immersion Trip Volunteer
Manage Contact
Create Mailing List
E-Mail
Calendar
To do / Tasks
This is the main menu navigation bar, from here you may access any of the interfaces.
This menu remains visible and accessible (and in the same place) throughout the user experience

Figure 1: Main Menu

Navigation Screens

Manage Contact



Accept Gift In Kind

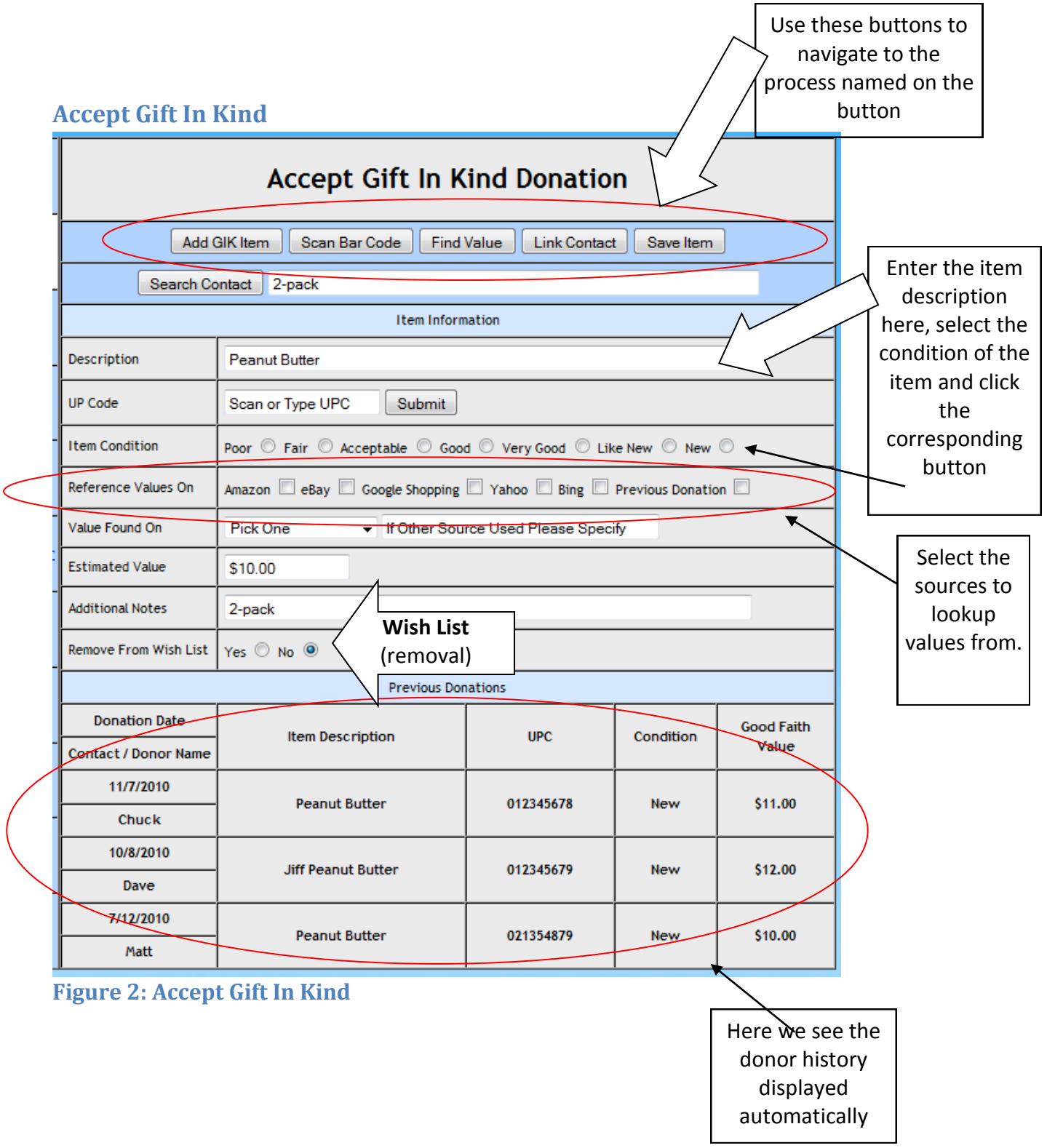


Figure 2: Accept Gift In Kind

Accept Monetary Donation

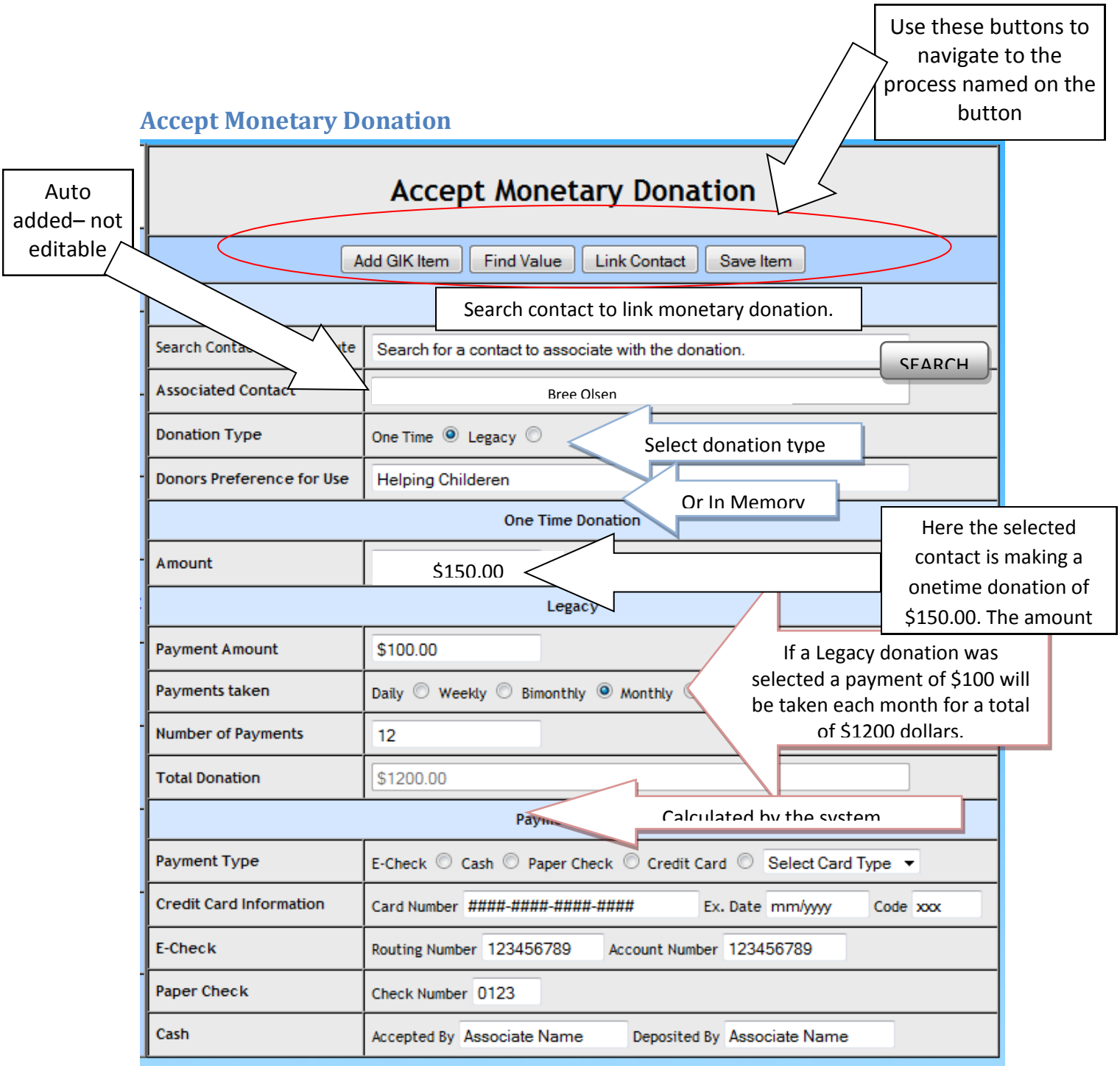


Figure 3: Accept Monetary Donation

The HHM Associate searches for a contact (Figure 5). If the contact does not exist, a new contact is created (Figure 2). Either "One Time" or "Legacy" donation is specified (Radio Button so only one at a time). If "One Time" is selected a lump sum is then entered. If legacy donation is selected then the payment amount, the frequency of the payment, and the number of payments is use by the system to calculate the total donation amount. How the payment is collected is the same for either type. After the donation is save, it becomes part of the linked donors History (Figure 9).

Contact Search Results

○	Bree	Olsen	123 Street Rd	Milford	MA	02038	555-3878
○	Carmella	Bing	132 Road Street	Millis	MA	01776	555-5580
○	Hanna	Hilton	3 Long Street	Medway	MA	02135	555-8249
○	Julia	Ann	999 Hard Street	Medfied	MA	02053	555-4031
○	Jaden	James	88 Soft Street	Medford	MA	02136	555-2694
○	Sara	Jay	342 Lexington Street	Middletown	MA	02485	555-4301
○	Sasha	Grey	8372 Cloud Street	Middleboro	MA	02135	555-2280
○							
○							
○							
○							
○							
○							

Figure 4: Contact Search Results

Create Mailing List

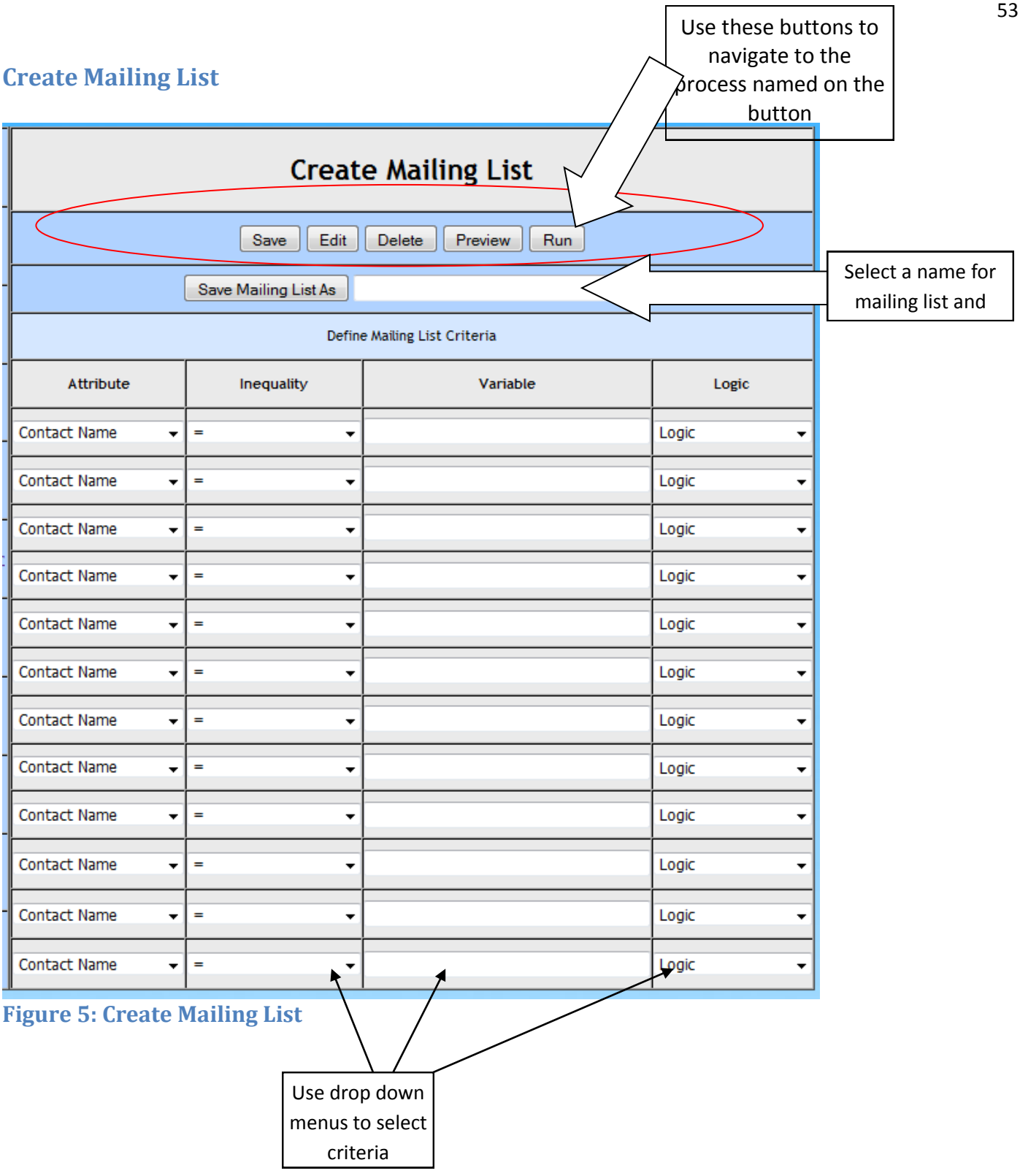
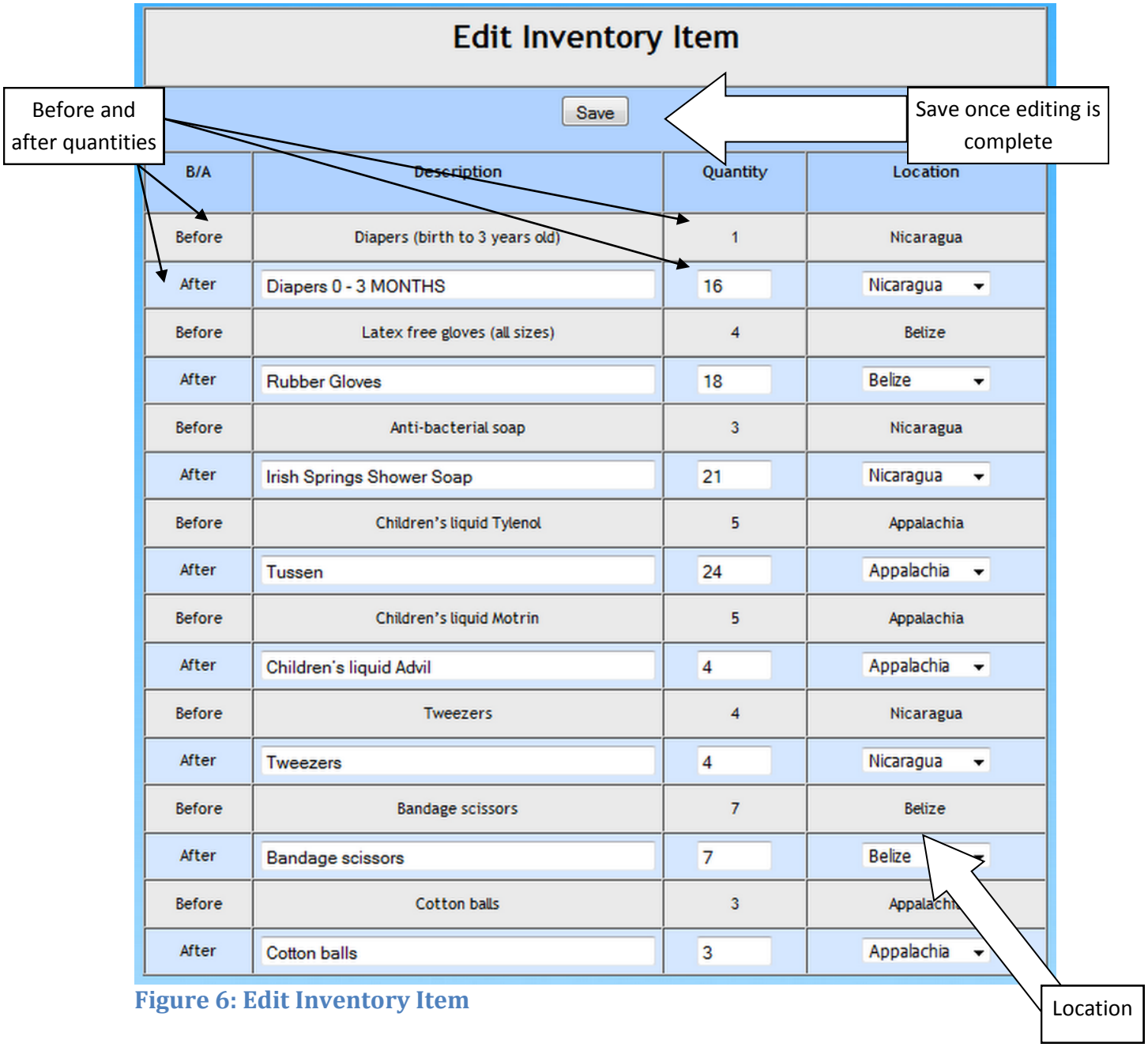


Figure 5: Create Mailing List

Edit Inventory Item



B/A	Description	Quantity	Location
Before	Diapers (birth to 3 years old)	1	Nicaragua
After	Diapers 0 - 3 MONTHS	16	Nicaragua
Before	Latex free gloves (all sizes)	4	Belize
After	Rubber Gloves	18	Belize
Before	Anti-bacterial soap	3	Nicaragua
After	Irish Springs Shower Soap	21	Nicaragua
Before	Children's liquid Tylenol	5	Appalachia
After	Tussen	24	Appalachia
Before	Children's liquid Motrin	5	Appalachia
After	Children's liquid Advil	4	Appalachia
Before	Tweezers	4	Nicaragua
After	Tweezers	4	Nicaragua
Before	Bandage scissors	7	Belize
After	Bandage scissors	7	Belize
Before	Cotton balls	3	Appalachi
After	Cotton balls	3	Appalachia

Figure 6: Edit Inventory Item

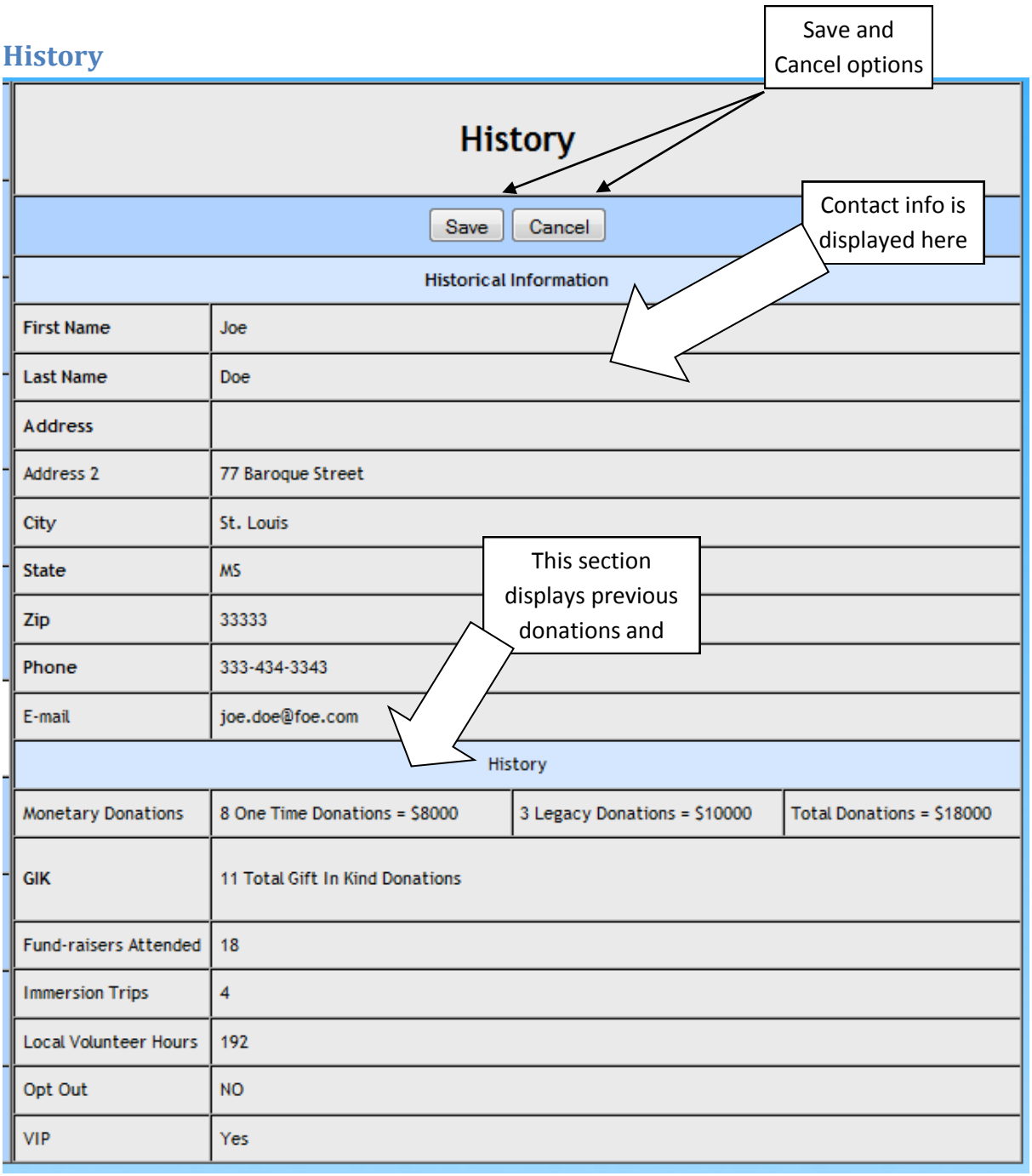
The In-kind Inventory form is the interface that allows users to view and update the inventory of supplies available. This prototype includes the primary form which users can add or remove items, assign the items to a region and update the quantity available.

E-Mail

Figure 7: E-Mail

This brings you to your commonly used e-mail client

History



History

Save Cancel

Historical Information

First Name	Joe
Last Name	Doe
Address	
Address 2	77 Baroque Street
City	St. Louis
State	MS
Zip	33333
Phone	333-434-3343
E-mail	joe.doe@foe.com

History

Monetary Donations	8 One Time Donations = $8000	3 Legacy Donations = $10000	Total Donations = $18000
GIK	11 Total Gift In Kind Donations		
Fund-raisers Attended	18		
Immersion Trips	4		
Local Volunteer Hours	192		
Opt Out	NO		
VIP	Yes		

Figure 8: History

Manage In Kind Inventory

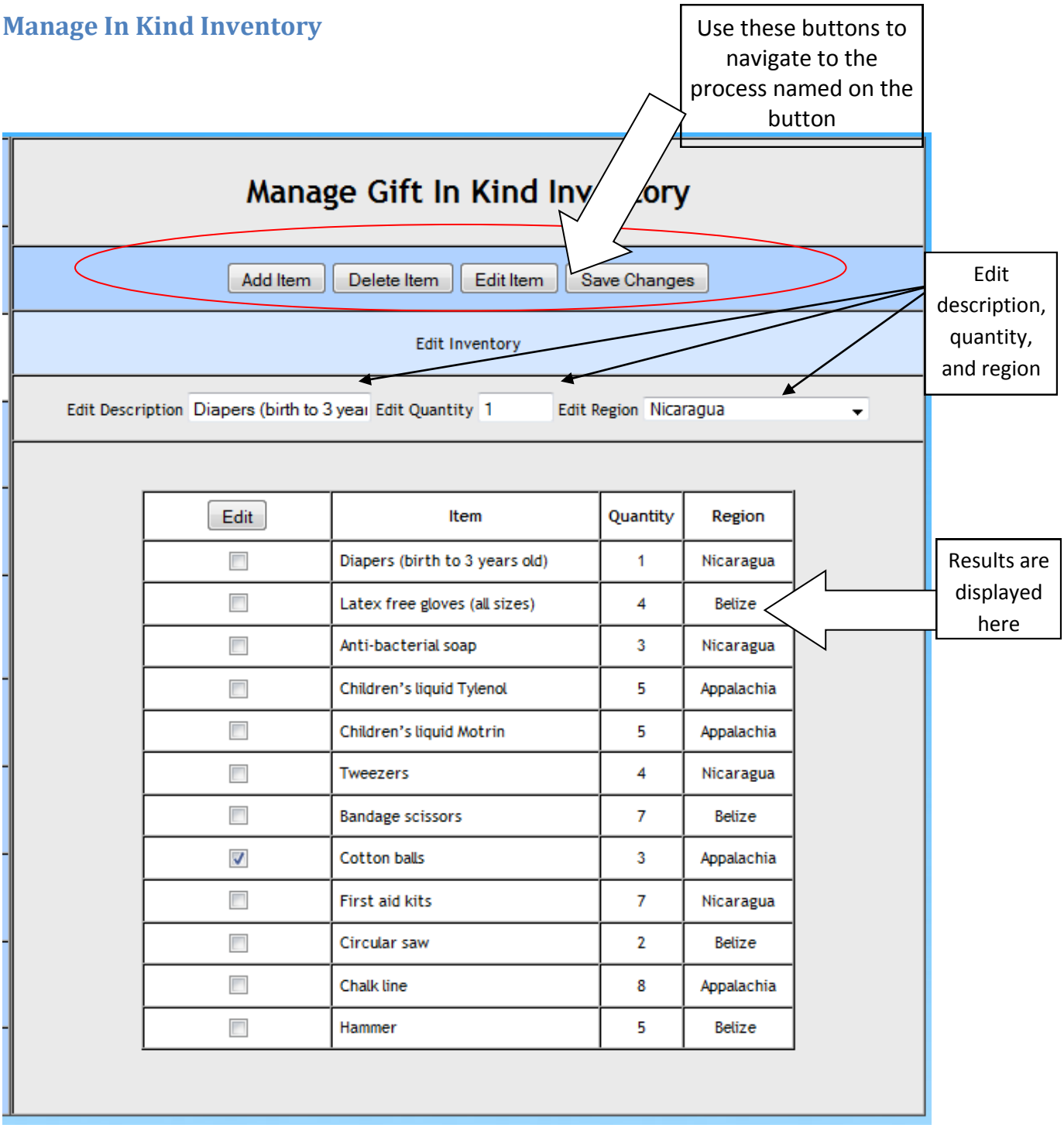


Figure 9: Manage In Kind Inventory

Manage Wish List

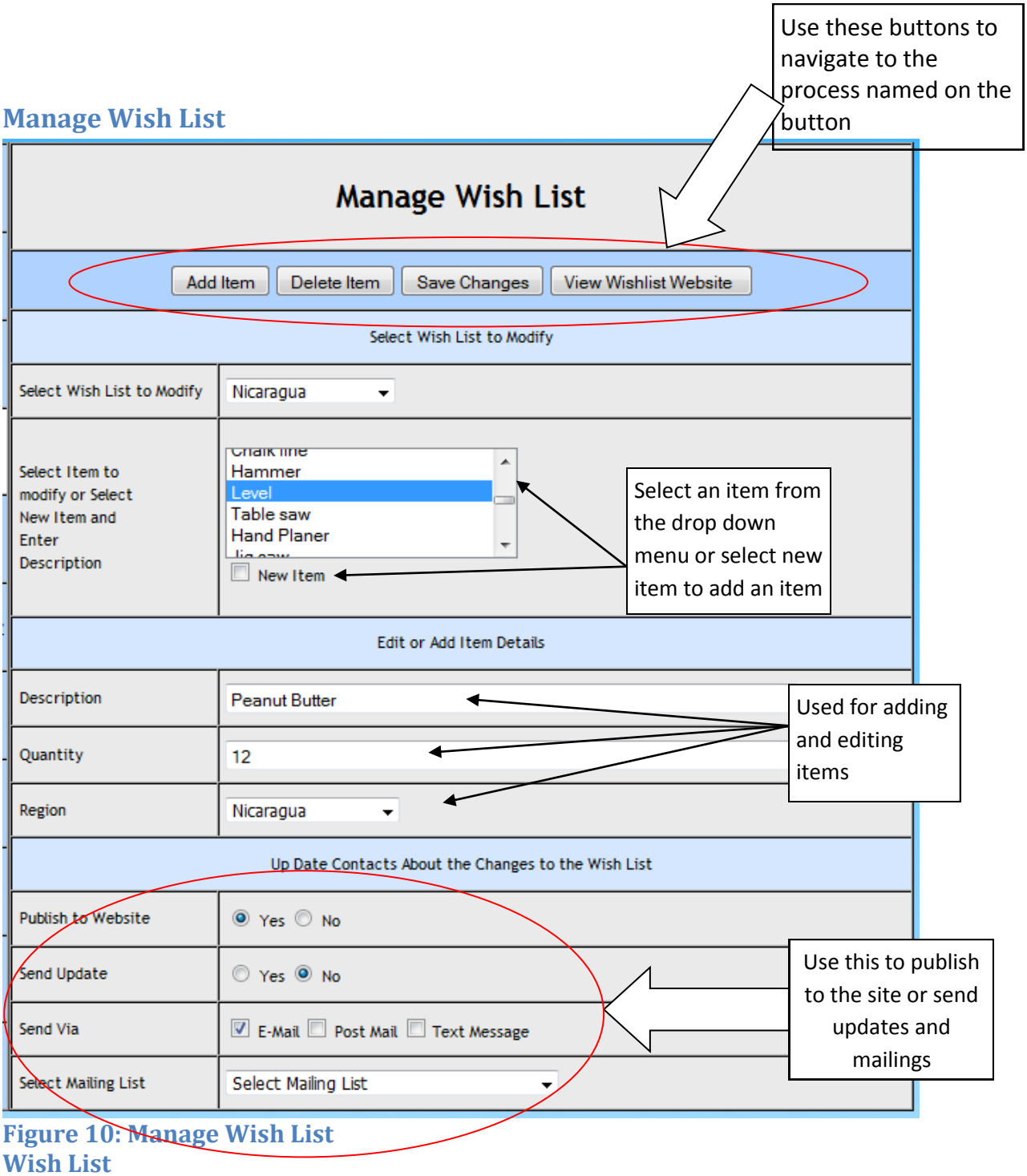


Figure 10: Manage Wish List

Wish List

The Wish List form is used to update the "Wish List" on the HHM website (the information from the wish list can be displayed in various applications). The wish list shows perspective donors specifically what is needed for the next mission trip. The web page Uses a bar graph to display the quantity that HHM has received in relation to how many they need without using actual numbers.

Merge Contact

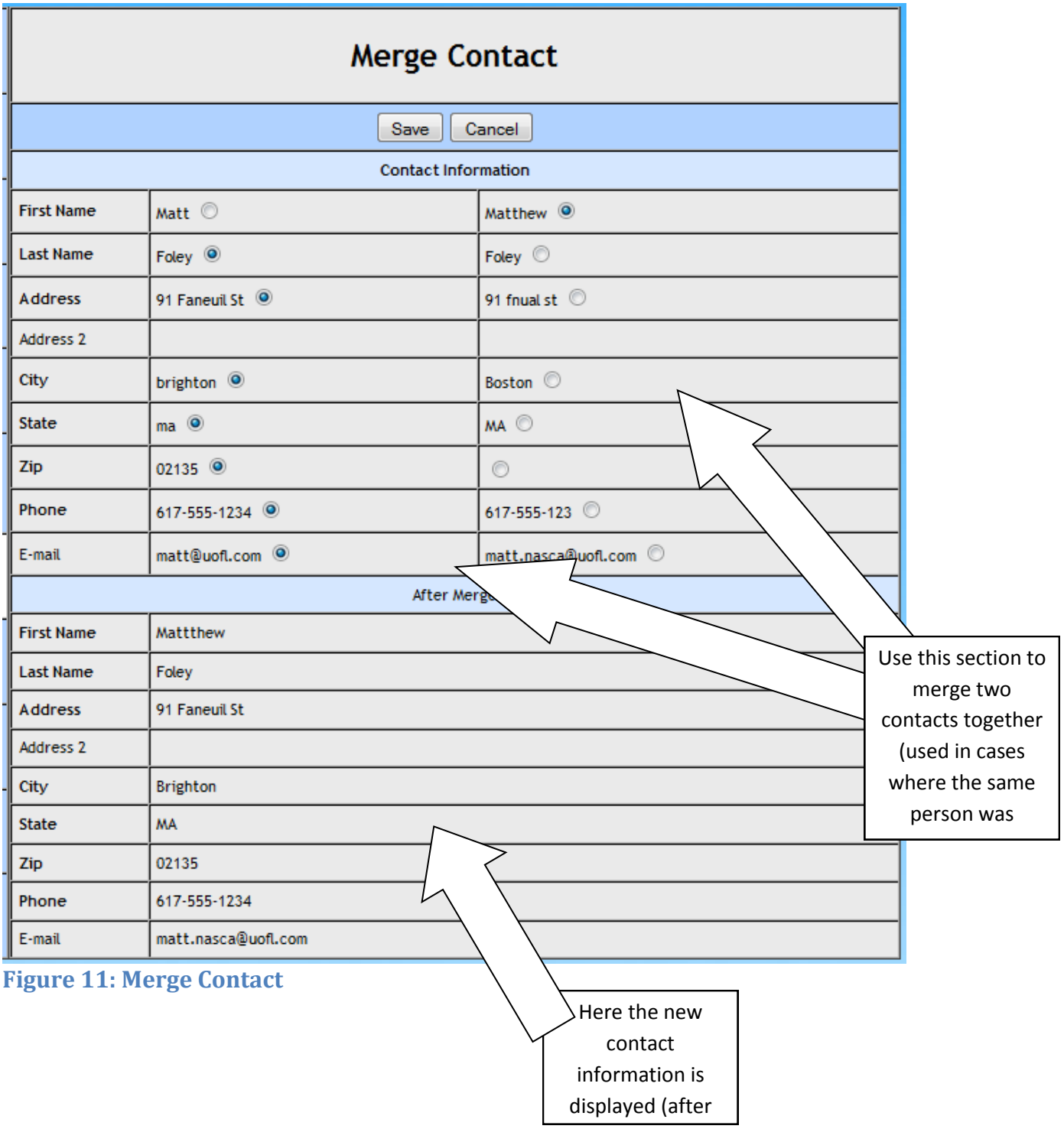


Figure 11: Merge Contact

To Do / Tasks

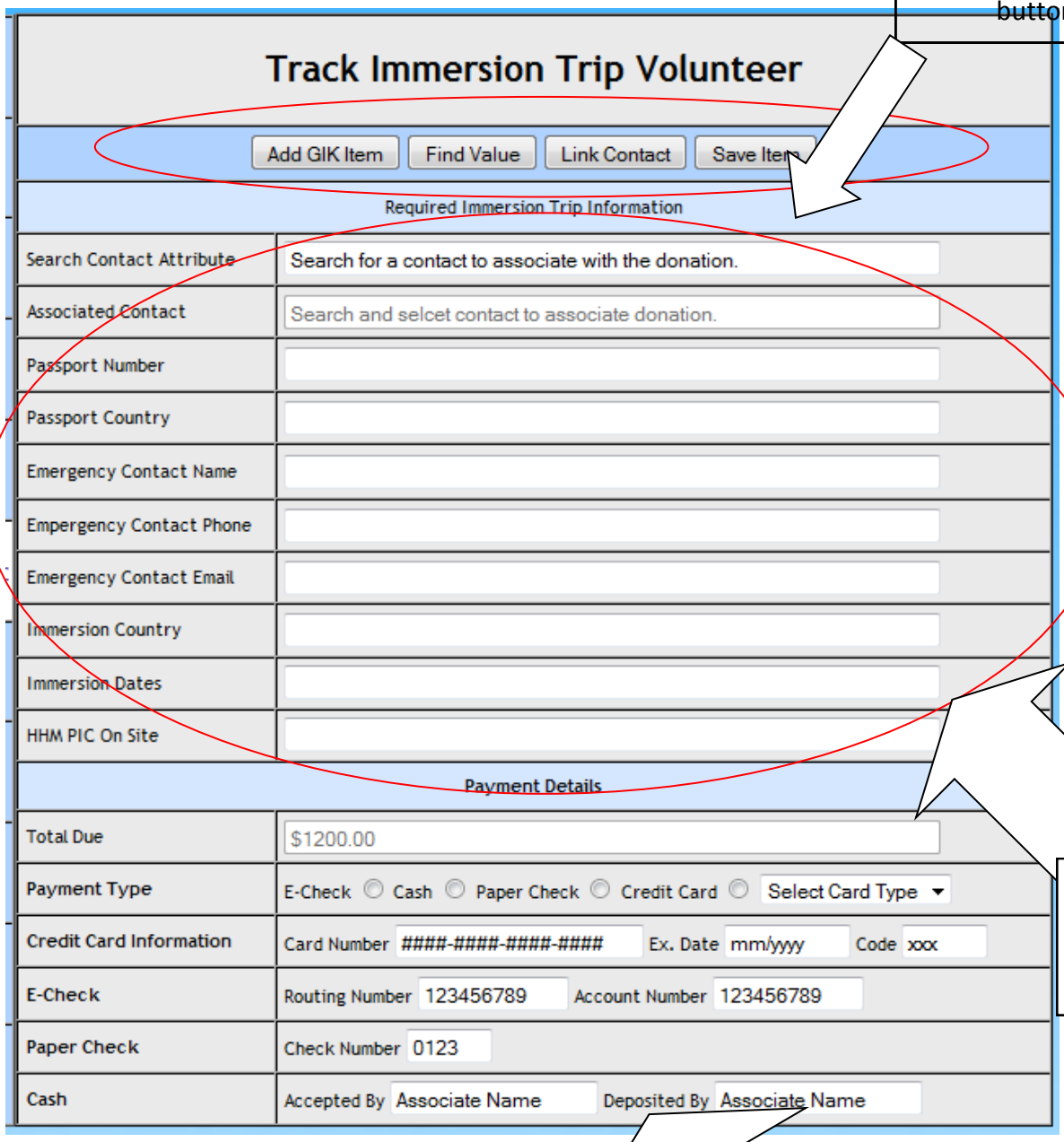
To-do / Tasks

Do it!			
Everyone			
Date	Description	Who	When
12/15/10	Help Someone	Volunteer	Everyday
12/15/10	Clean Something	Volunteer	Tomorrow
On Going	Buy Steve Lunch	Everyone	Everyday
12/11/10	HHM Presentation	Matt	NOW
Mine			
12/11/10	HHM Presentation	Matt	NOW

Figure 12: To Do / Tasks

This form displays the to-do tasks that are on the agenda

Track Immersion Trip Volunteer



Use these buttons to navigate to the process named on the button

These fields are required for immersion trip volunteers.

This section at the bottom contains the payment transaction area. Enter relevant credit card, cash, or eCheck information. All immersion trip costs are treated as monetary donations

Figure 13: Track Immersion Trip Volunteer

Track Local Volunteer

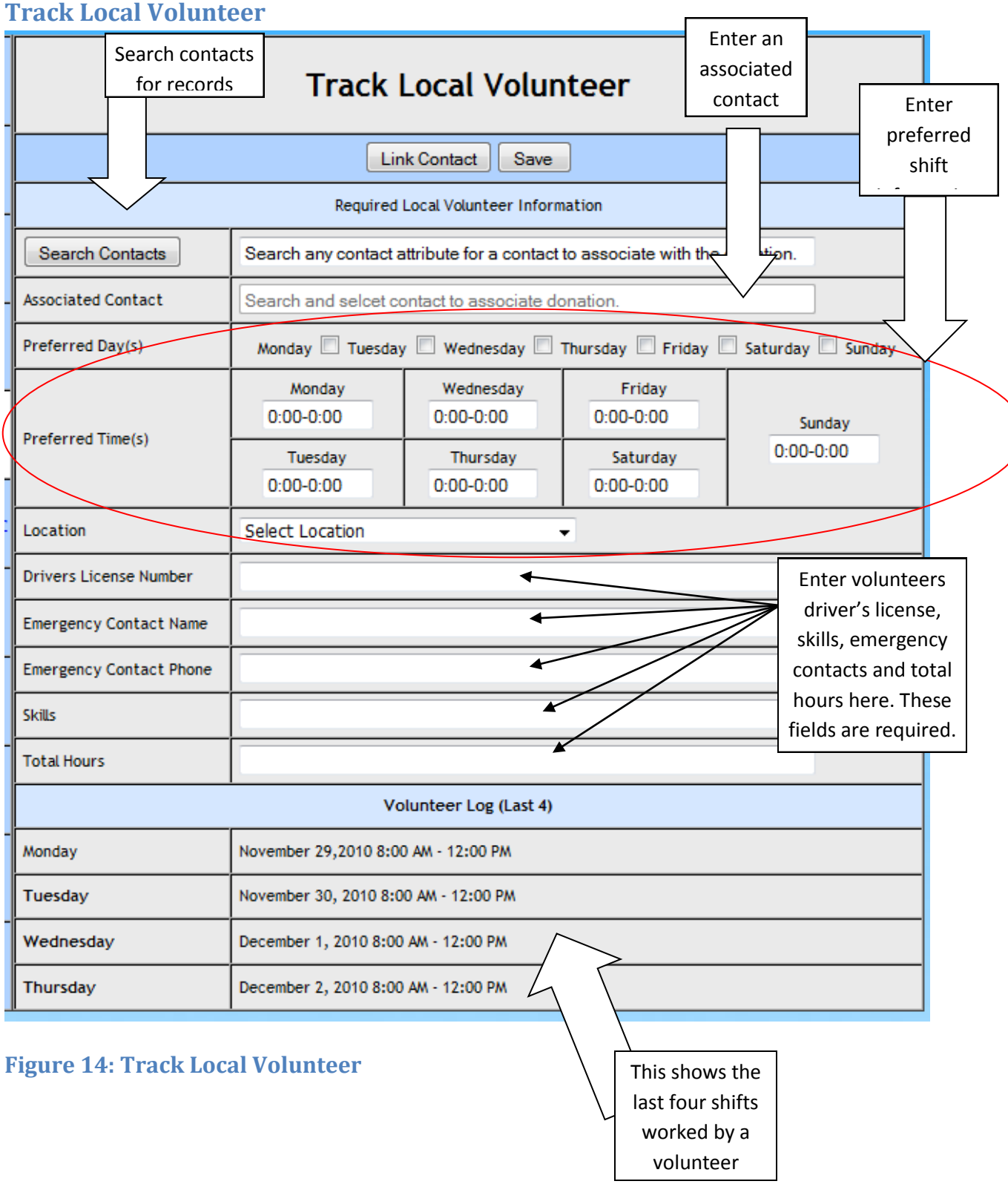


Figure 14: Track Local Volunteer

Calendar

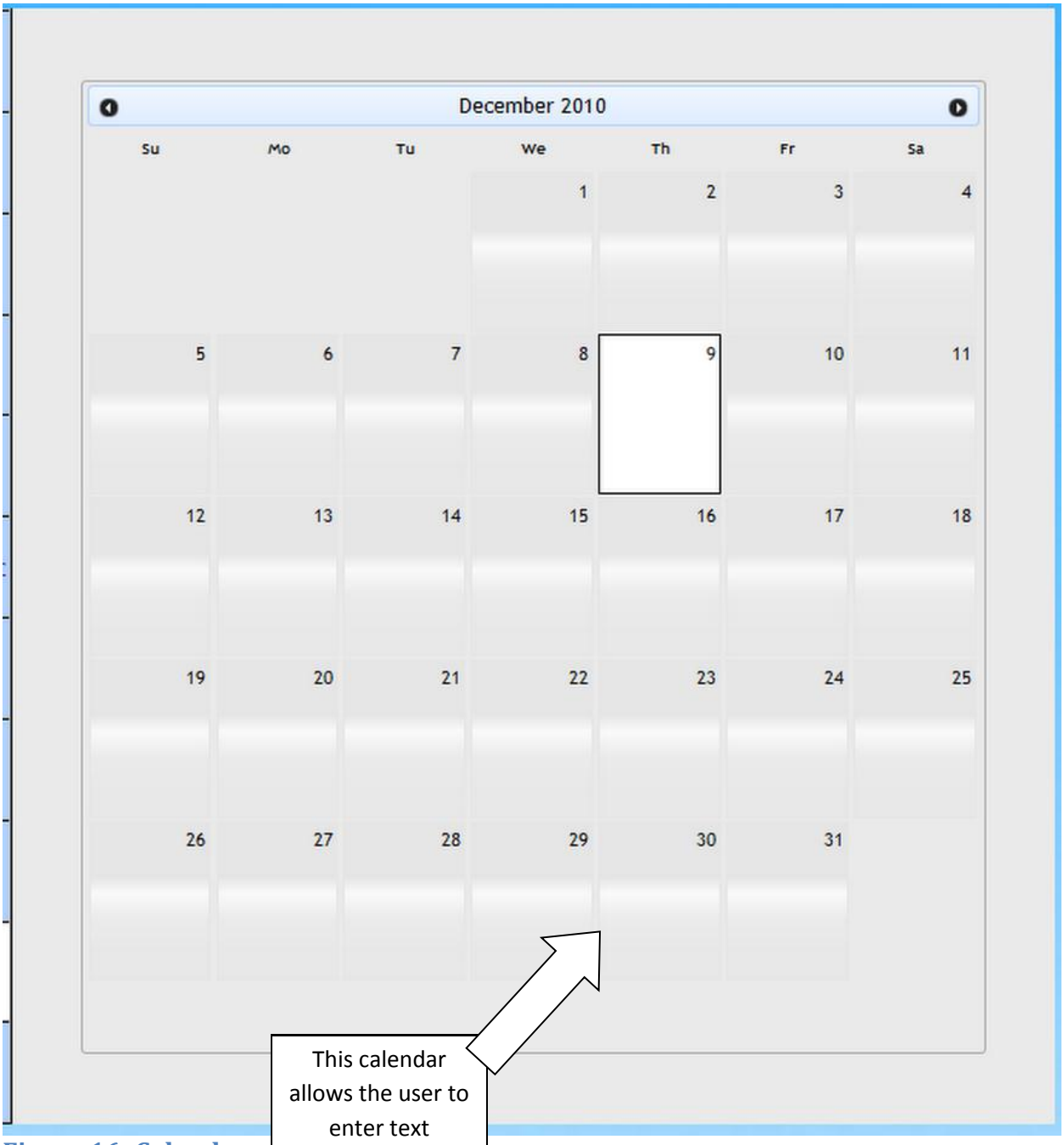
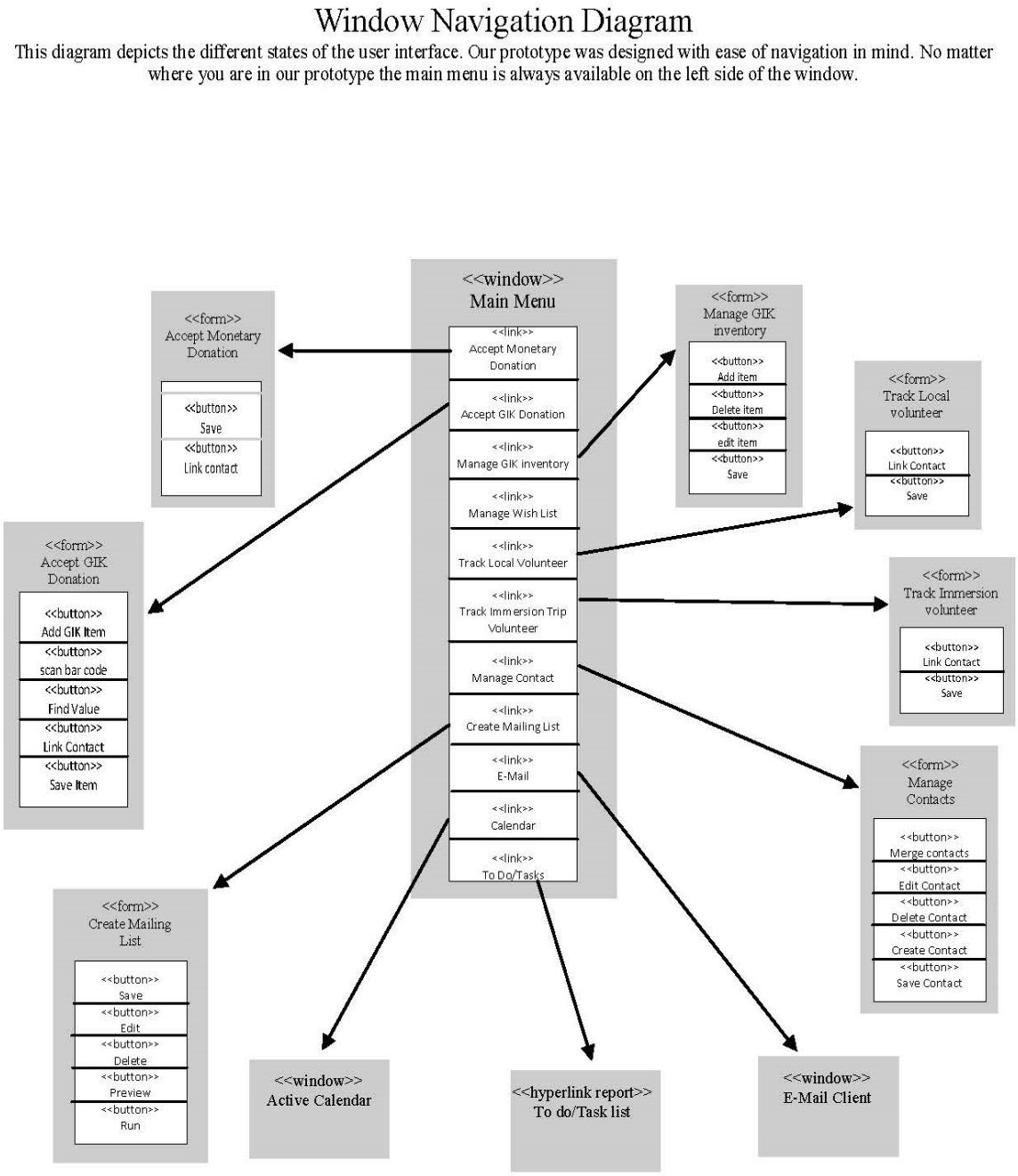
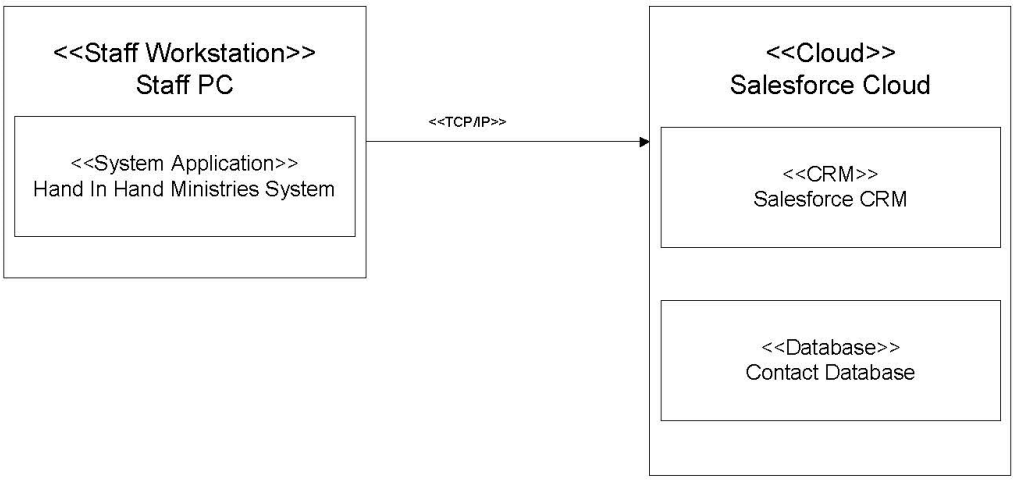


Figure 16: Calendar

Windows Navigation Diagram



Deployment Diagram



Client Narrative

The actual application runs on the cloud so the staff can access it anywhere with an Internet enabled device.

1. The staff accesses the Salesforce application through the PC terminal.
2. The computer is connected to the cloud through the Internet.
3. The cloud contains the Salesforce application and the database of contacts.

Nonfunctional Requirements

Nonfunctional Requirements

- The system shall use only one contact table that will be shared by all programs in the database.

Performance requirements

- The computer system shall have access to the Internet.
- The computer system shall be able to process minimum business processes ex. Outlook

Security requirements

- The system shall be capable of a minimum of 128 bit encryption.
- The system shall be secure.

Salesforce Security Policy

The Salesforce CRM provides already built in security to ensure that the database integrity is never compromised. The Salesforce production equipment is located in California at a facility that provides 24-hour physical security, palm print and picture identification systems, redundant electrical generators, redundant data center air conditioners, and other backup equipment designed to keep servers continually up and running. Salesforce uses the strongest encryption products to protect customer data and communications, including 128-bit VeriSign SSL Certification. Salesforce also provides access to a geographically remote disaster recovery facility, along with required hardware, software, and Internet connectivity. In the event the Salesforce production facilities were to be rendered unavailable.

Nonfunctional Requirements

1. Operational Requirements

1.1 The system shall use only one contact table that will be shared by all programs and databases.

1.2 The system shall be compatible with software that is currently in use (i.e. Microsoft Excel)

1.3 The system shall operate in either Windows or Mac environments.

1.4 The system will work over Web environment with any web browser.

2. Performance Requirements

2.1 The system shall be fully operational 95% of the time.

2.2 The system shall retrieve donor records with an average response time of 5 seconds or less.

2.3 The system shall lookup values and return the results with an average response time of 5 seconds or less.

2.4 The contact database will be updated in real time.

3. Hardware Requirements

3.1 The system shall use personal Computers (Mac or PC) manufactured after 2005 with a minimum of 1Gb RAM.

3.2 The system shall have broadband access, routers, modem, and service.

3.3 The system shall be equipped with either Wi-Fi or Ethernet cards to interact with the donor database.

3.4 The system shall use small I.D. scanners (one per donation site)

3.5 The system shall use UPC scanners (minimum one per donation site)

3.6 The system shall have Ink-Jet or Laser printers (a minimum of one per site)

4. Software Requirements

4.1 The system shall run on Windows XP SP3/Windows Vista/Windows 7 or Mac OSX.

5. Security Requirements

5.1 The system shall be capable of a minimum of 128 bit encryption between HIHM data and the Salesforce database.

5.2 Any user accessing the system from a mobile device will be required to authenticate before gaining access to the system.

5.3 All files uploaded to any HIHM computer terminal will be automatically scanned for viruses before being executed.

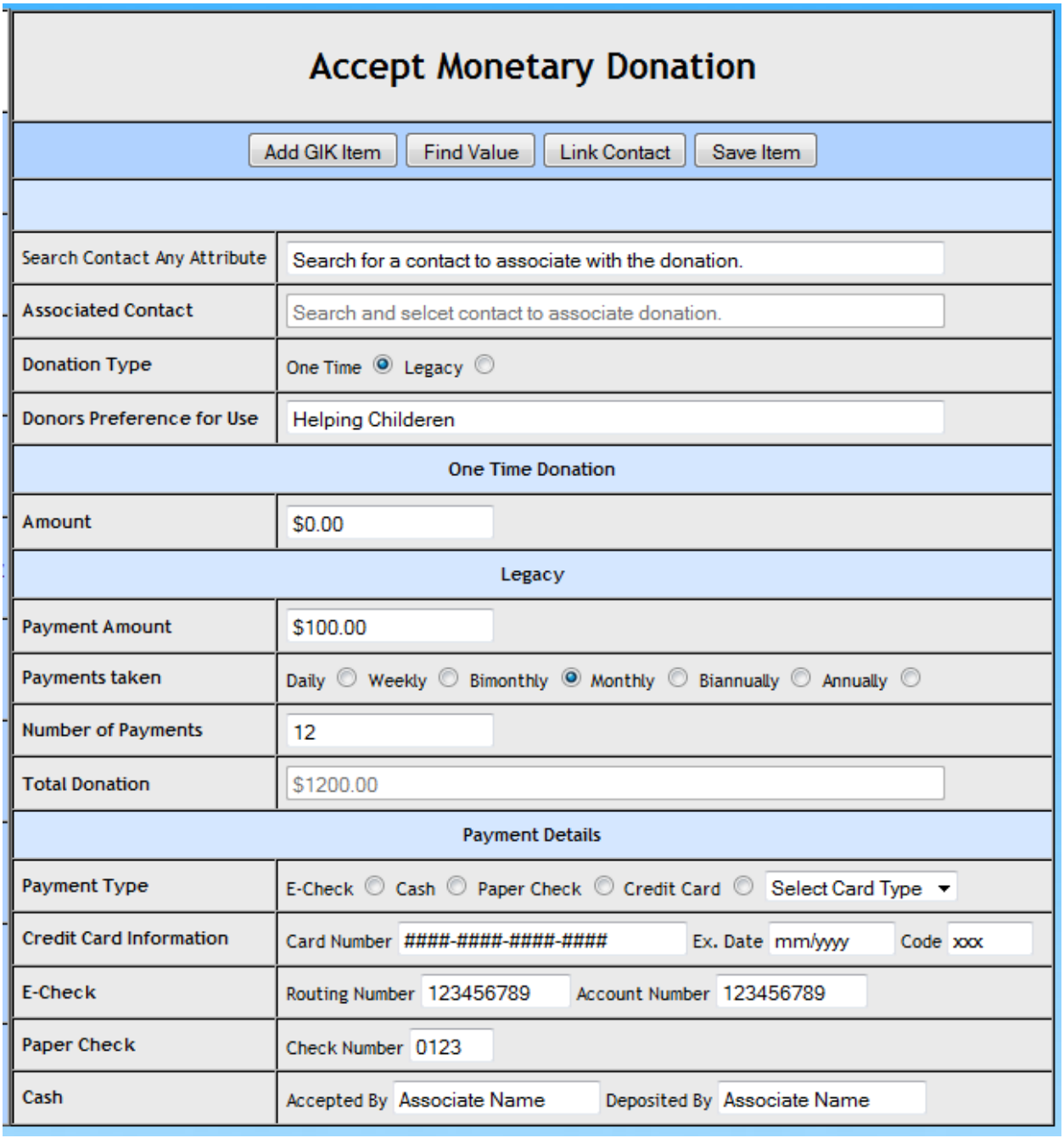
Hand in Hand Security Policy

As mentioned in the nonfunctional requirements the staff will have access to the system through any web browser including a mobile web browser. The user will have to authenticate employee information every time they log into the system regardless of the browser. The HIHM computers will automatically scan any downloaded file for viruses before executing to ensure data security and protect any private information.

Salesforce Security Policy

The Salesforce CRM provides already built in security to ensure that the database integrity is never compromised. The Salesforce production equipment is located in California at a facility that provides 24-hour physical security, palm print and picture identification systems, redundant electrical generators, redundant data center air conditioners, and other backup equipment designed to keep servers continually up and running. Salesforce uses the strongest encryption products to protect customer data and communications, including 128-bit VeriSign SSL Certification. Salesforce also provides access to a geographically remote disaster recovery facility, along with required hardware, software, and Internet connectivity. In the event the Salesforce production facilities were to be rendered unavailable.

Prototype


Accept Monetary Donation
Add GIK Item
Find Value
Link Contact
Save Item
Search Contact Any Attribute
Search for a contact to associate with the donation.
Associated Contact
Search and selcet contact to associate donation.
Donation Type
One Time
Legacy
Donors Preference for Use
Helping Childeren
One Time Donation
Amount
$0.00
Legacy
Payment Amount
$100.00
Payments taken
Daily
Weekly
Bimonthly
Monthly
Biannually
Annually
Number of Payments
12
Total Donation
$1200.00
Payment Details
Payment Type
E-Check
Cash
Paper Check
Credit Card
Select Card Type
Credit Card Information
Card Number
####-####-####-####
Ex. Date
mm/yyyy
Code
xxx
E-Check
Routing Number
123456789
Account Number
123456789
Paper Check
Check Number
0123
Cash
Accepted By
Associate Name
Deposited By
Associate Name

Accept Gift In Kind Donation

Add GIK Item | Scan Bar Code | Find Value | Link Contact | Save Item

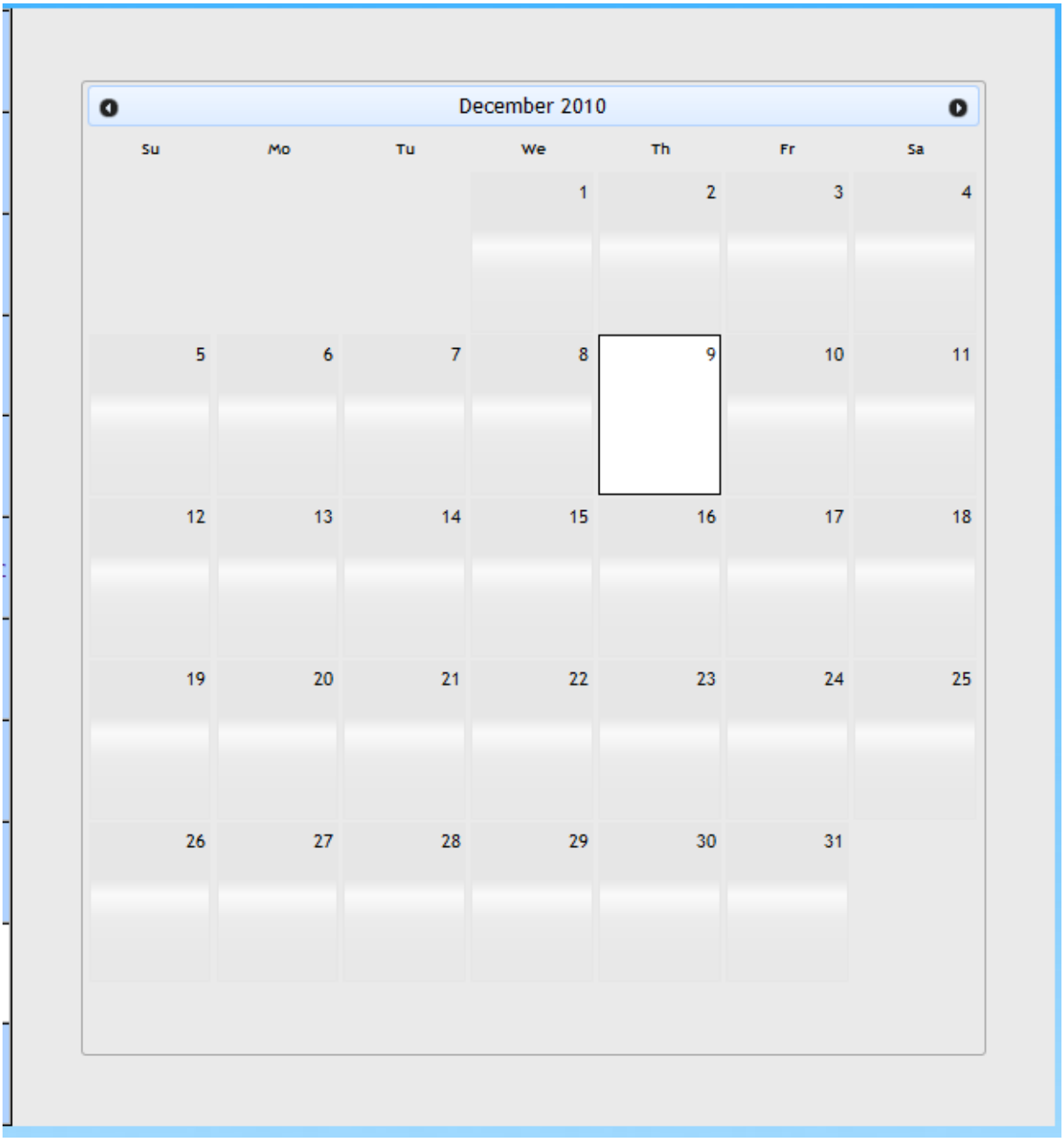
Search Contact: 2-pack

Item Information

Description	Peanut Butter
UP Code	Scan or Type UPC [Submit]
Item Condition	Poor ○ Fair ○ Acceptable ○ Good ○ Very Good ○ Like New ○ New ○
Reference Values On	Amazon ☐ eBay ☐ Google Shopping ☐ Yahoo ☐ Bing ☐ Previous Donation ☐
Value Found On	Pick One ▾ If Other Source Used Please Specify
Estimated Value	$10.00
Additional Notes	2-pack
Remove From Wish List	Yes ○ No ◉

Previous Donations

Donation Date / Contact / Donor Name	Item Description	UPC	Condition	Good Faith Value
11/7/2010 / Chuck	Peanut Butter	012345678	New	$11.00
10/8/2010 / Dave	Jiff Peanut Butter	012345679	New	$12.00
7/12/2010 / Matt	Peanut Butter	021354879	New	$10.00


December 2010
Su
Mo
Tu
We
Th
Fr
Sa
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31

Create Mailing List
Save
Edit
Delete
Preview
Run
Save Mailing List As
Define Mailing List Criteria
Attribute
Inequality
Variable
Logic
Contact Name
=
Logic

Edit Inventory Item

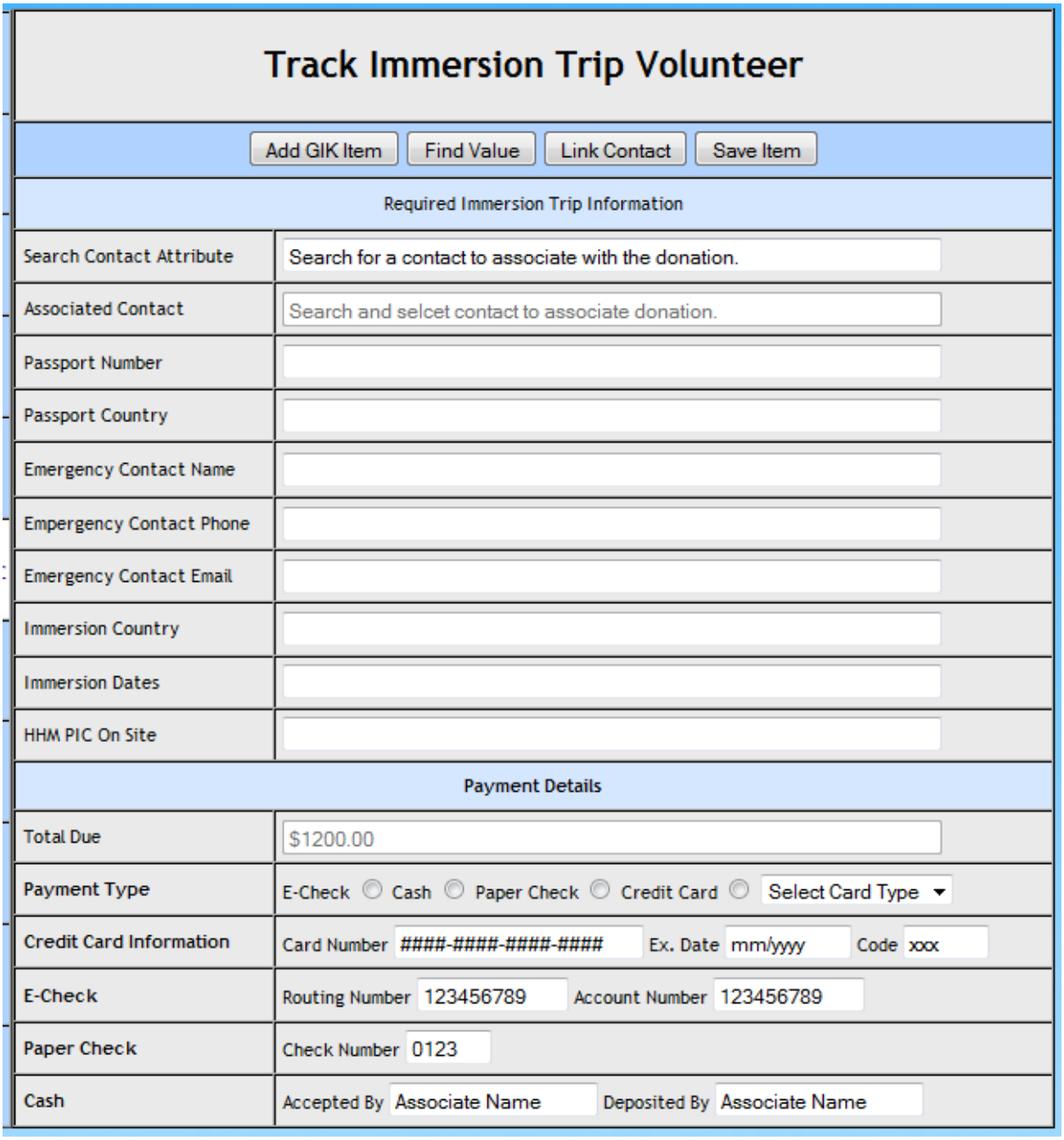
Save

B/A	Description	Quantity	Location
Before	Diapers (birth to 3 years old)	1	Nicaragua
After	Diapers 0 - 3 MONTHS	16	Nicaragua
Before	Latex free gloves (all sizes)	4	Belize
After	Rubber Gloves	18	Belize
Before	Anti-bacterial soap	3	Nicaragua
After	Irish Springs Shower Soap	21	Nicaragua
Before	Children's liquid Tylenol	5	Appalachia
After	Tussen	24	Appalachia
Before	Children's liquid Motrin	5	Appalachia
After	Children's liquid Advil	4	Appalachia
Before	Tweezers	4	Nicaragua
After	Tweezers	4	Nicaragua
Before	Bandage scissors	7	Belize
After	Bandage scissors	7	Belize
Before	Cotton balls	3	Appalachia
After	Cotton balls	3	Appalachia

History

Save Cancel

Historical Information			
First Name	Joe		
Last Name	Doe		
Address			
Address 2	77 Baroque Street		
City	St. Louis		
State	MS		
Zip	33333		
Phone	333-434-3343		
E-mail	joe.doe@foe.com		
History			
Monetary Donations	8 One Time Donations = $8000	3 Legacy Donations = $10000	Total Donations = $18000
GIK	11 Total Gift In Kind Donations		
Fund-raisers Attended	18		
Immersion Trips	4		
Local Volunteer Hours	192		
Opt Out	NO		
VIP	Yes		



Track Immersion Trip Volunteer

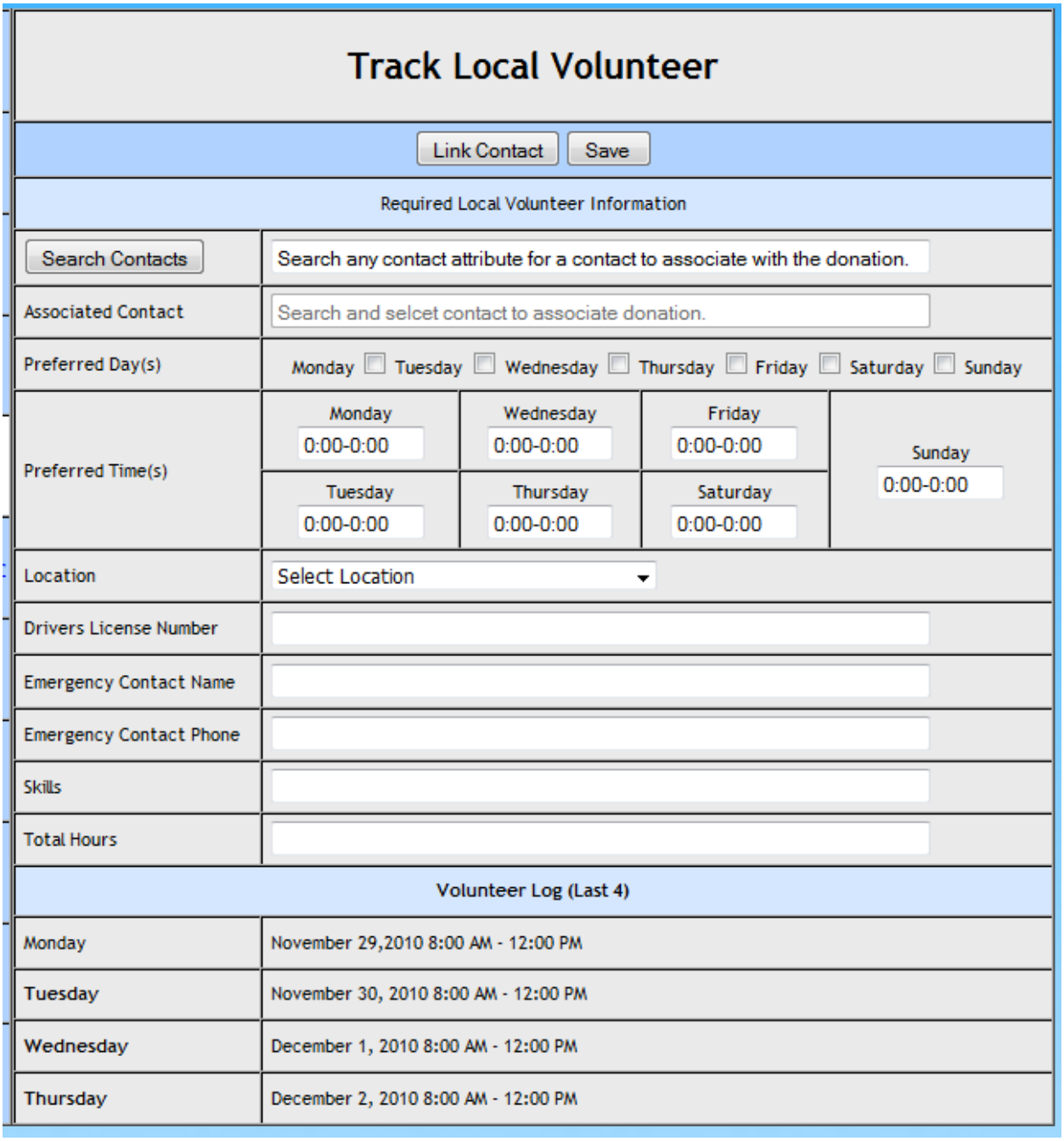
Add GIK Item | Find Value | Link Contact | Save Item

Required Immersion Trip Information

Search Contact Attribute	Search for a contact to associate with the donation.
Associated Contact	Search and selcet contact to associate donation.
Passport Number	
Passport Country	
Emergency Contact Name	
Empergency Contact Phone	
Emergency Contact Email	
Immersion Country	
Immersion Dates	
HHM PIC On Site	

Payment Details

Total Due	$1200.00
Payment Type	E-Check ○ Cash ○ Paper Check ○ Credit Card ○ Select Card Type ▾
Credit Card Information	Card Number ####-####-####-#### Ex. Date mm/yyyy Code xxx
E-Check	Routing Number 123456789 Account Number 123456789
Paper Check	Check Number 0123
Cash	Accepted By Associate Name Deposited By Associate Name



Track Local Volunteer

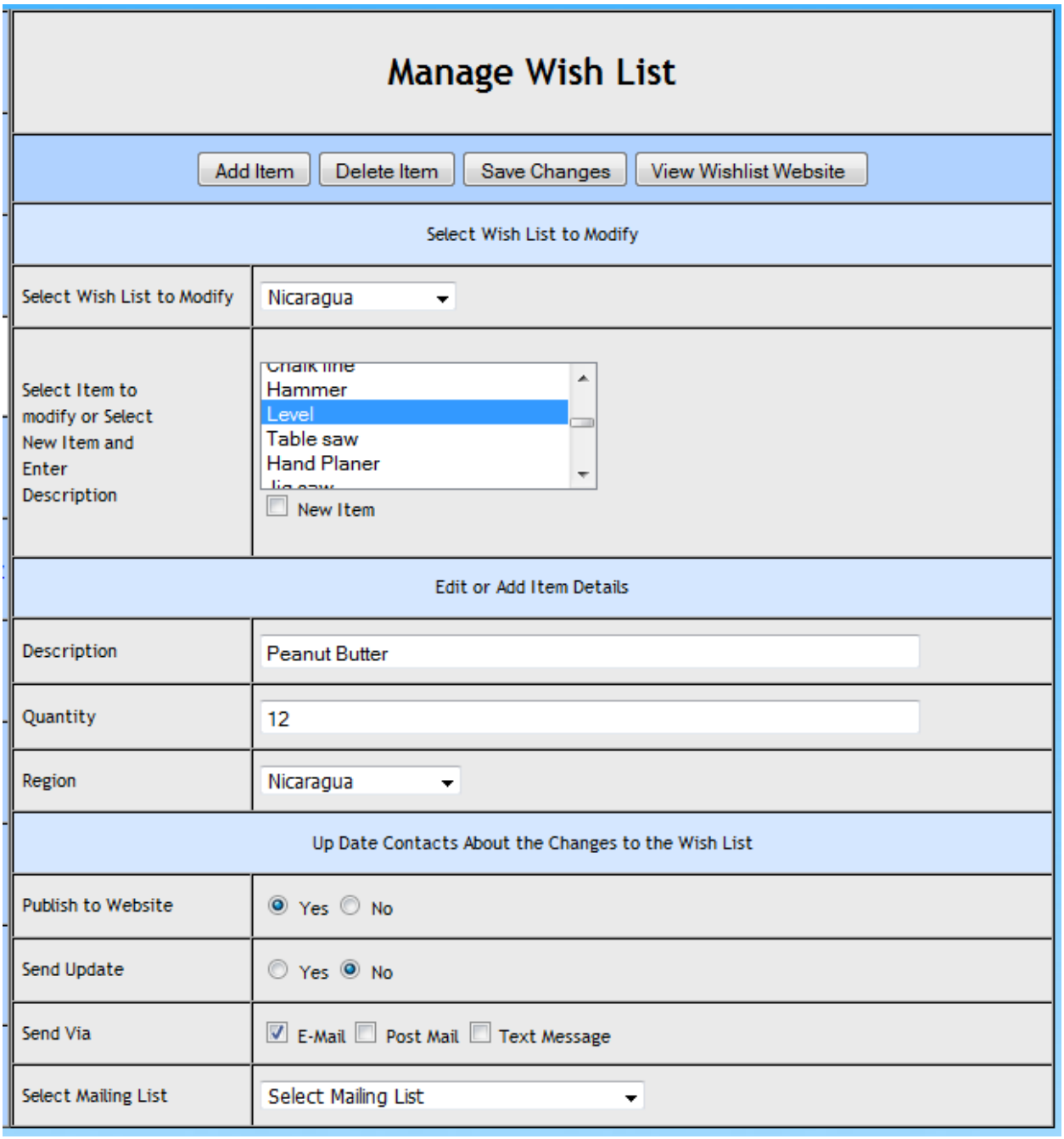
Link Contact | Save

Required Local Volunteer Information

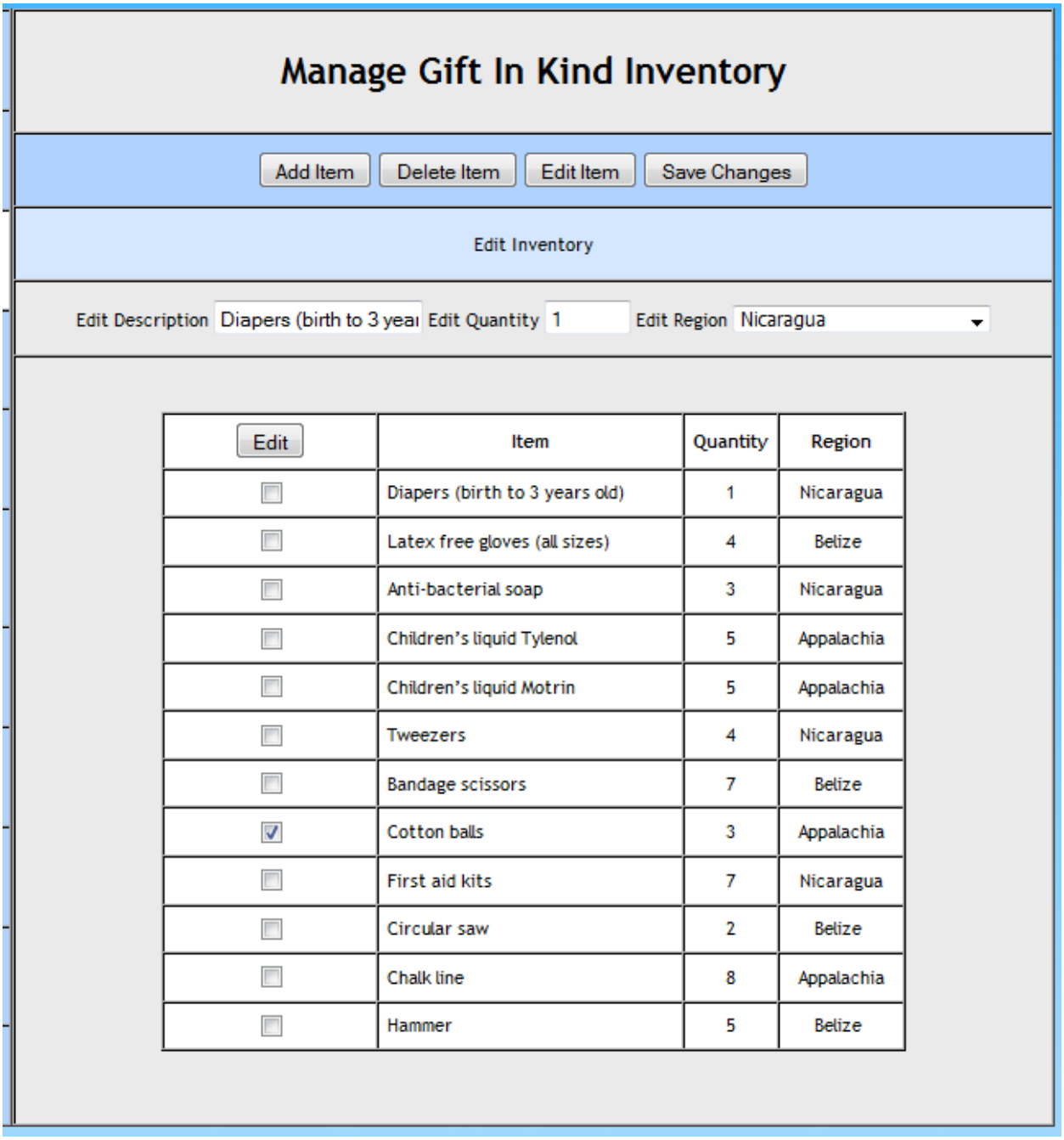
Search Contacts	Search any contact attribute for a contact to associate with the donation.
Associated Contact	Search and selcet contact to associate donation.
Preferred Day(s)	Monday ☐ Tuesday ☐ Wednesday ☐ Thursday ☐ Friday ☐ Saturday ☐ Sunday
Preferred Time(s)	Monday 0:00-0:00 · Tuesday 0:00-0:00 · Wednesday 0:00-0:00 · Thursday 0:00-0:00 · Friday 0:00-0:00 · Saturday 0:00-0:00 · Sunday 0:00-0:00
Location	Select Location
Drivers License Number	
Emergency Contact Name	
Emergency Contact Phone	
Skills	
Total Hours	

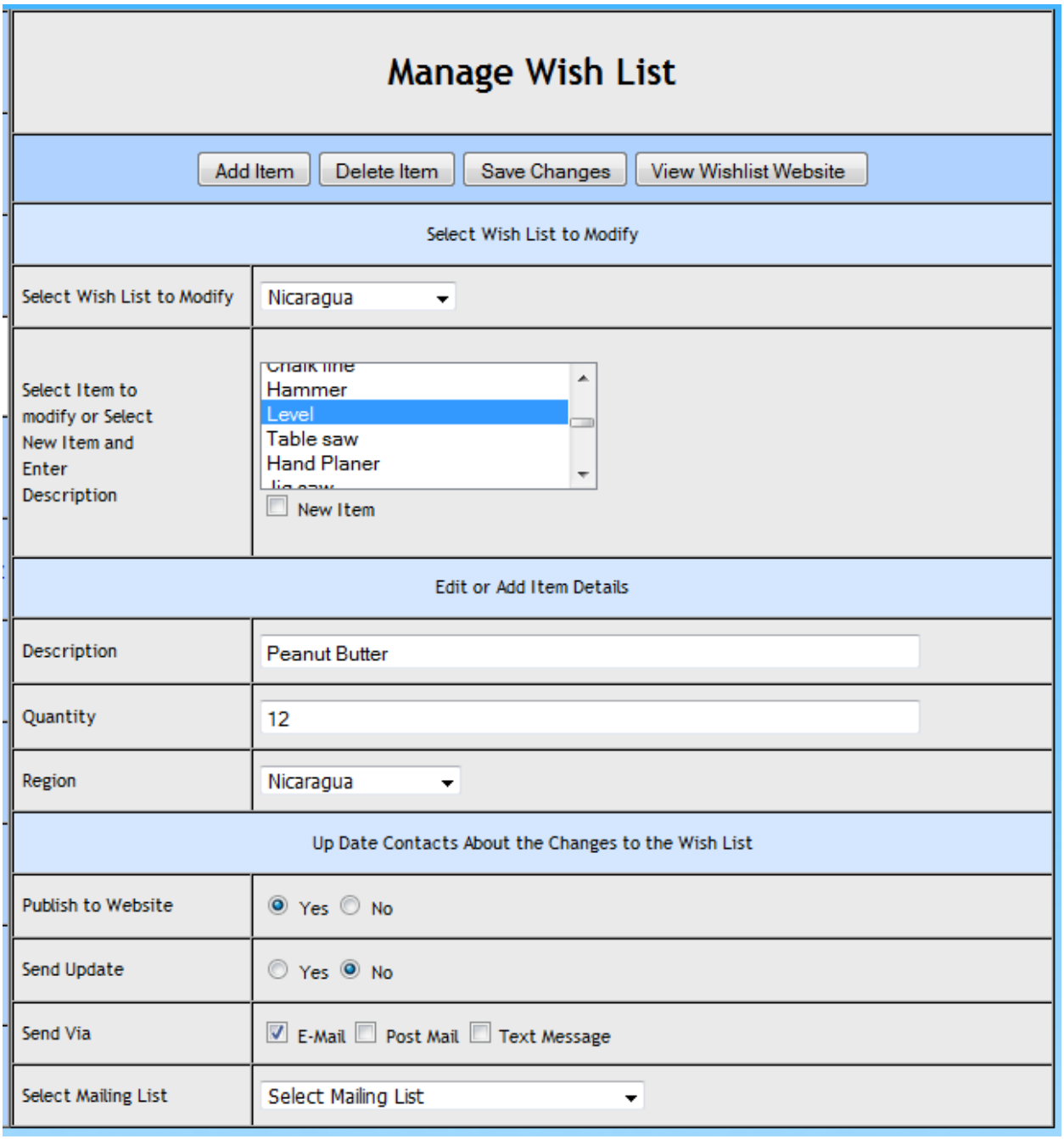
Volunteer Log (Last 4)

Monday	November 29,2010 8:00 AM - 12:00 PM
Tuesday	November 30, 2010 8:00 AM - 12:00 PM
Wednesday	December 1, 2010 8:00 AM - 12:00 PM
Thursday	December 2, 2010 8:00 AM - 12:00 PM


Manage Wish List
Add Item
Delete Item
Save Changes
View Wishlist Website
Select Wish List to Modify
Select Wish List to Modify
Nicaragua
Select Item to modify or Select New Item and Enter Description
Chalk line
Hammer
Level
Table saw
Hand Planer
Jig saw
New Item
Edit or Add Item Details
Description
Peanut Butter
Quantity
12
Region
Nicaragua
Up Date Contacts About the Changes to the Wish List
Publish to Website
Yes
No
Send Update
Yes
No
Send Via
E-Mail
Post Mail
Text Message
Select Mailing List
Select Mailing List

Manage Contact
Merge Contacts
Edit Contact
Delete Contact
Create Contact
Save Contact
Search Contacts
Contact Information
First Name
Chuck
Last Name
Wilson
Address
University of Louisville
Address 2
COB
City
Louisville
State
KY
Zip
00000-0000
Phone
502-123-4567
E-mail
cc@uofl.com
Birthday
mm/dd/yyyy
Historical Activity
VIP
Yes
No


Manage Gift In Kind Inventory
Add Item
Delete Item
Edit Item
Save Changes
Edit Inventory
Edit Description
Diapers (birth to 3 yeai
Edit Quantity
1
Edit Region
Nicaragua
Edit
Item
Quantity
Region
Diapers (birth to 3 years old) 1 Nicaragua
Latex free gloves (all sizes) 4 Belize
Anti-bacterial soap 3 Nicaragua
Children's liquid Tylenol 5 Appalachia
Children's liquid Motrin 5 Appalachia
Tweezers 4 Nicaragua
Bandage scissors 7 Belize
Cotton balls 3 Appalachia
First aid kits 7 Nicaragua
Circular saw 2 Belize
Chalk line 8 Appalachia
Hammer 5 Belize


Manage Wish List
Add Item
Delete Item
Save Changes
View Wishlist Website
Select Wish List to Modify
Select Wish List to Modify
Nicaragua
Select Item to modify or Select New Item and Enter Description
Chalk line
Hammer
Level
Table saw
Hand Planer
Jig saw
New Item
Edit or Add Item Details
Description
Peanut Butter
Quantity
12
Region
Nicaragua
Up Date Contacts About the Changes to the Wish List
Publish to Website
Yes
No
Send Update
Yes
No
Send Via
E-Mail
Post Mail
Text Message
Select Mailing List
Select Mailing List

Merge Contact

Save Cancel

Contact Information		
First Name	Matt ○	Matthew ◉
Last Name	Foley ◉	Foley ○
Address	91 Faneuil St ◉	91 fnual st ○
Address 2		
City	brighton ◉	Boston ○
State	ma ◉	MA ○
Zip	02135 ◉	○
Phone	617-555-1234 ◉	617-555-123 ○
E-mail	matt@uofl.com ◉	matt.nasca@uofl.com ○

After Merge	
First Name	Mattthew
Last Name	Foley
Address	91 Faneuil St
Address 2	
City	Brighton
State	MA
Zip	02135
Phone	617-555-1234
E-mail	matt.nasca@uofl.com

Contact Search Results

Select Contact to Associate or Edit

Select | Search again with new criteria.

Select

○	Bree	Olsen	123 Street Rd	Milford	MA	02038	555-3878
○	Carmella	Bing	132 Road Street	Millis	MA	01776	555-5580
○	Hanna	Hilton	3 Long Street	Medway	MA	02135	555-8249
○	Julia	Ann	999 Hard Street	Medfied	MA	02053	555-4031
○	Jaden	James	88 Soft Street	Medford	MA	02136	555-2694
○	Sara	Jay	342 Lexington Street	Middletown	MA	02485	555-4301
○	Sasha	Grey	8372 Cloud Street	Middleboro	MA	02135	555-2280
○							
○							
○							
○							
○							
○							

To-do / Tasks

Do it!			
Everyone			
Date	Description	Who	When
12/15/10	Help Someone	Volunteer	Everyday
12/15/10	Clean Something	Volunteer	Tomorrow
On Going	Buy Steve Lunch	Everyone	Everyday
12/11/10	HHM Presentation	Matt	NOW
Mine			
12/11/10	HHM Presentation	Matt	NOW

West Louisville Community Ministries

Beta Test: 1

Beta testing: using the illustrated instructions and the activity diagram

Welcome to Beta testing! During this stage we will be providing you with working beta versions of your salesforce CRM (client relations management) database. With each new iteration we will add functionality that you can experiment with. In this way you will be able to get a “hands on” feel for the system and make suggestions based on your experiences.

Going through this process is very simple. On the next page you will find an activity diagram. This diagram shows the workflow (as we see it) for a particular process. Each step on the diagram is represented by an oval with a description inside, and each of those ovals exists with a lane. These lanes represent the people who perform the activity. In the pages following the activity diagram, you will find illustrated instructions for performing the activity. Each oval in the activity diagram will appear above each illustrated step so that you can easily compare which step you are performing with its position on the diagram.

This particular test is designed to test client and appointment objects. The instructions will take you through creating a client and setting an appointment for that client. It will also go through editing a client to add spouse and other information.

Note: Searching for an existing contact is not included in this test, we are examining ways to do this. We hope to include that functionality in such a way that it is automatic.

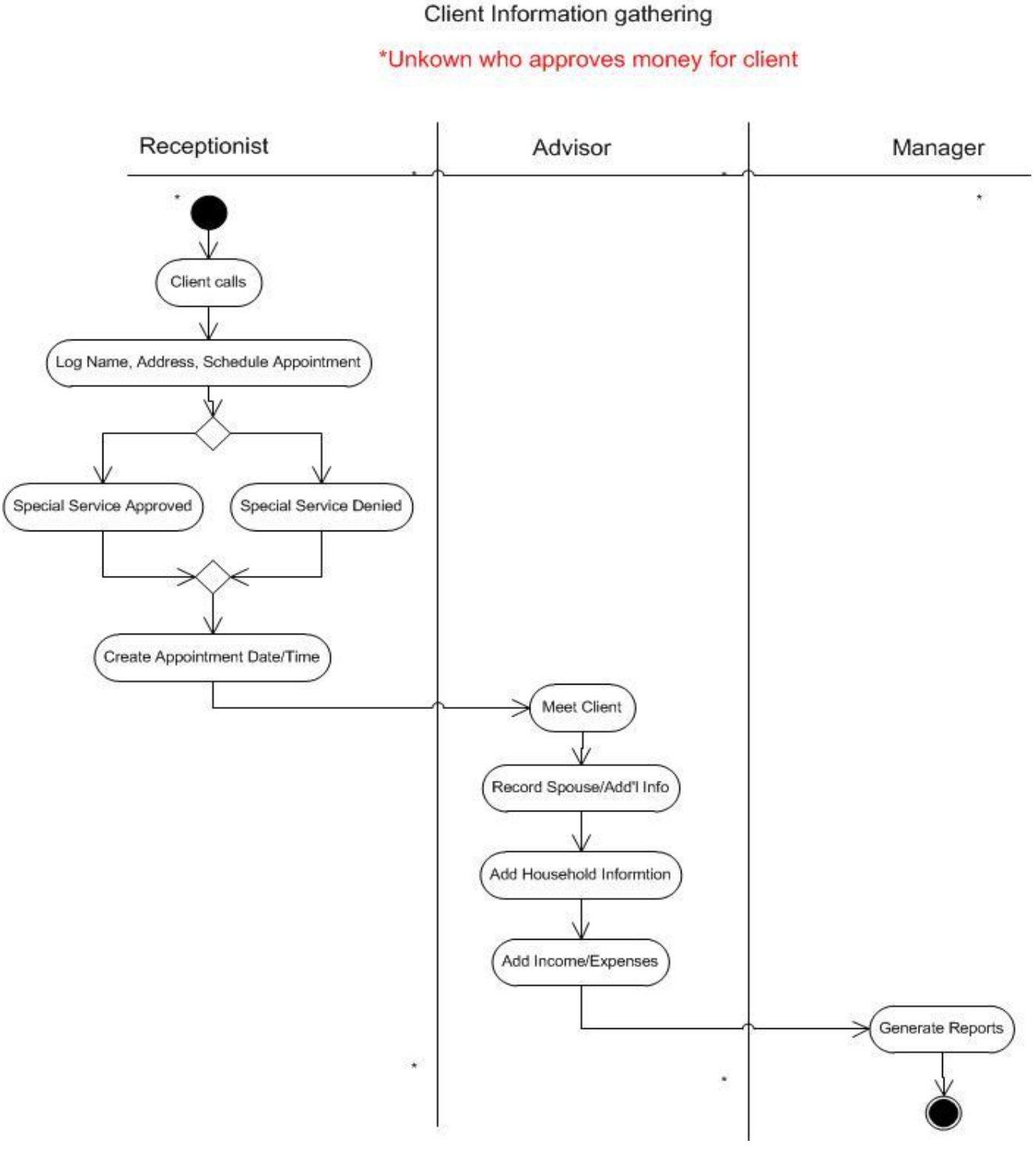
To log into Salesforce, go to Salesforce.com and enter the following username and password:

Jocylen: Username: Jocylen@betatest.com
Password: tester#1

Ms. Flowers: Username: MsFlowers@betatest.com
Password: tester#2

Please log in with your provided username and password, this will assist us in further testing The scheduling and appointment parts of the system.

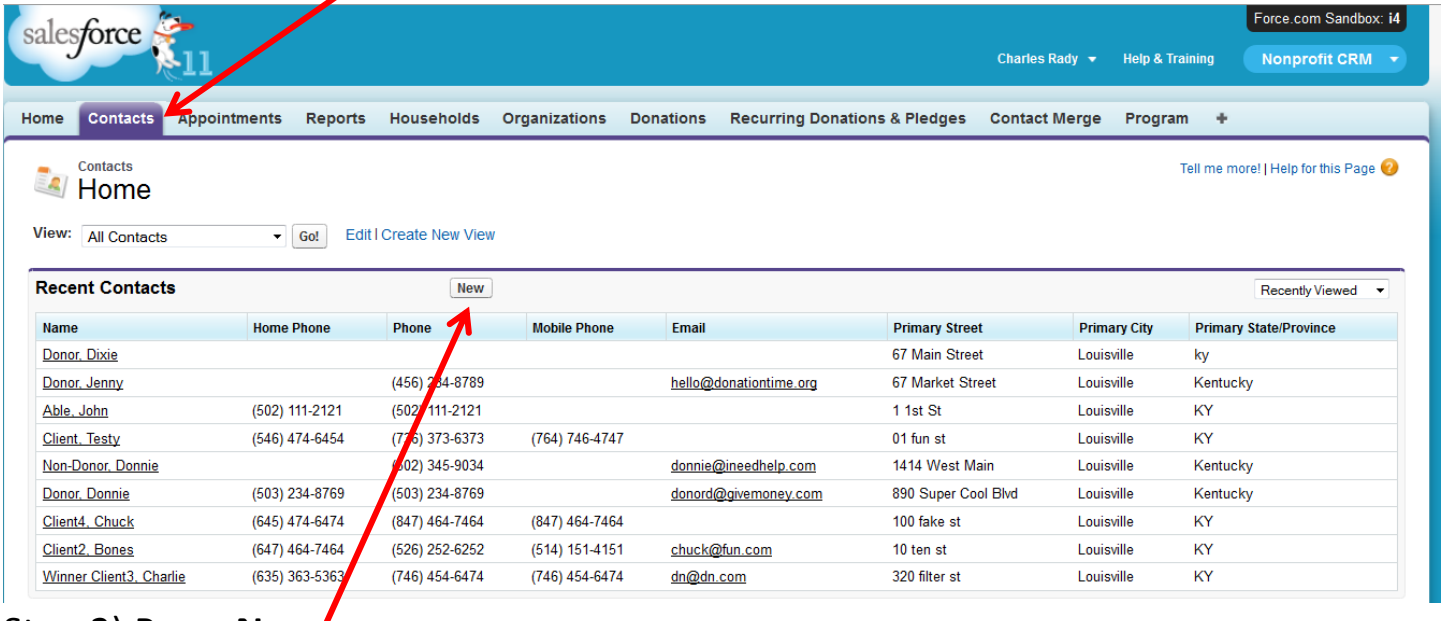
Activity Diagram



Client calls

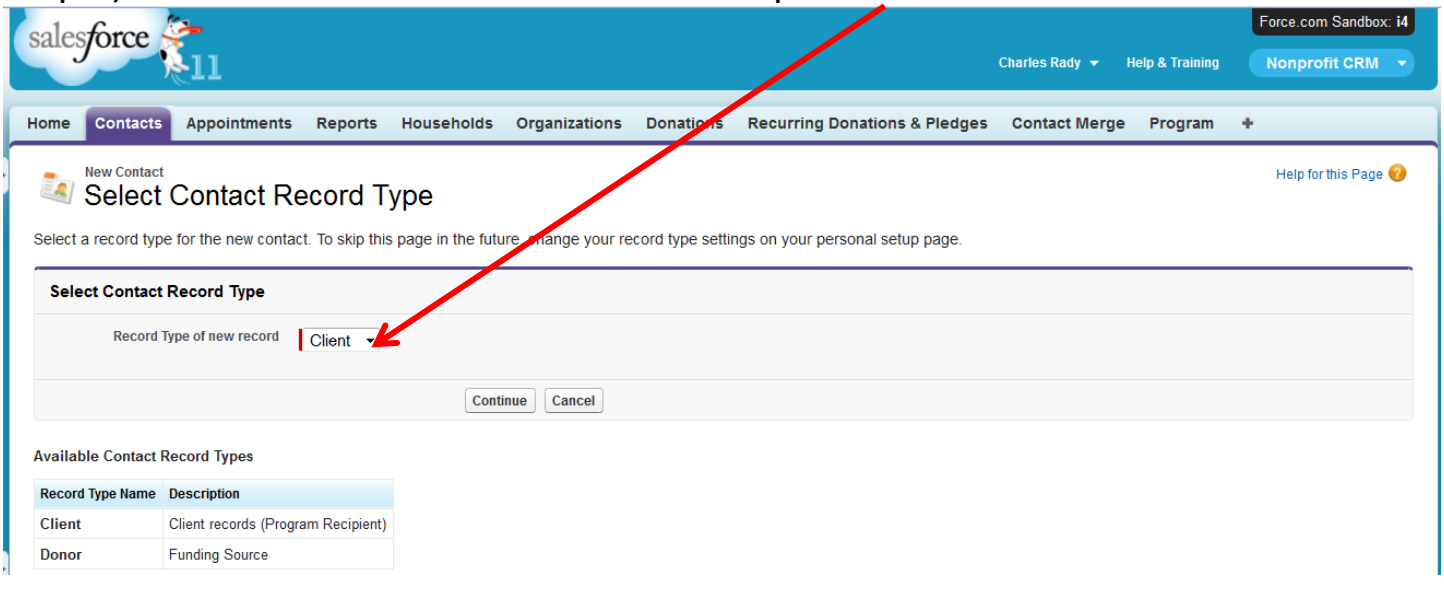
A client calls in and asks for an appointment

Step 1) Press the **Contacts** tab


Force.com Sandbox: i4
Charles Rady
Help & Training
Nonprofit CRM
Home
Contacts
Appointments
Reports
Households
Organizations
Donations
Recurring Donations & Pledges
Contact Merge
Program
Contacts
Home
Tell me more! | Help for this Page
View: All Contacts Go! Edit | Create New View
Recent Contacts
New
Recently Viewed
Name | Home Phone | Phone | Mobile Phone | Email | Primary Street | Primary City | Primary State/Province
Donor, Dixie | | | | | 67 Main Street | Louisville | ky
Donor, Jenny | | (456) 234-8789 | | hello@donationtime.org | 67 Market Street | Louisville | Kentucky
Able, John | (502) 111-2121 | (502) 111-2121 | | | 1 1st St | Louisville | KY
Client, Testy | (546) 474-6454 | (736) 373-6373 | (764) 746-4747 | | 01 fun st | Louisville | KY
Non-Donor, Donnie | | (502) 345-9034 | | donnie@ineedhelp.com | 1414 West Main | Louisville | Kentucky
Donor, Donnie | (503) 234-8769 | (503) 234-8769 | | donord@givemoney.com | 890 Super Cool Blvd | Louisville | Kentucky
Client4, Chuck | (645) 474-6474 | (847) 464-7464 | (847) 464-7464 | | 100 fake st | Louisville | KY
Client2, Bones | (647) 464-7464 | (526) 252-6252 | (514) 151-4151 | chuck@fun.com | 10 ten st | Louisville | KY
Winner Client3, Charlie | (635) 363-5363 | (746) 454-6474 | (746) 454-6474 | dn@dn.com | 320 filter st | Louisville | KY

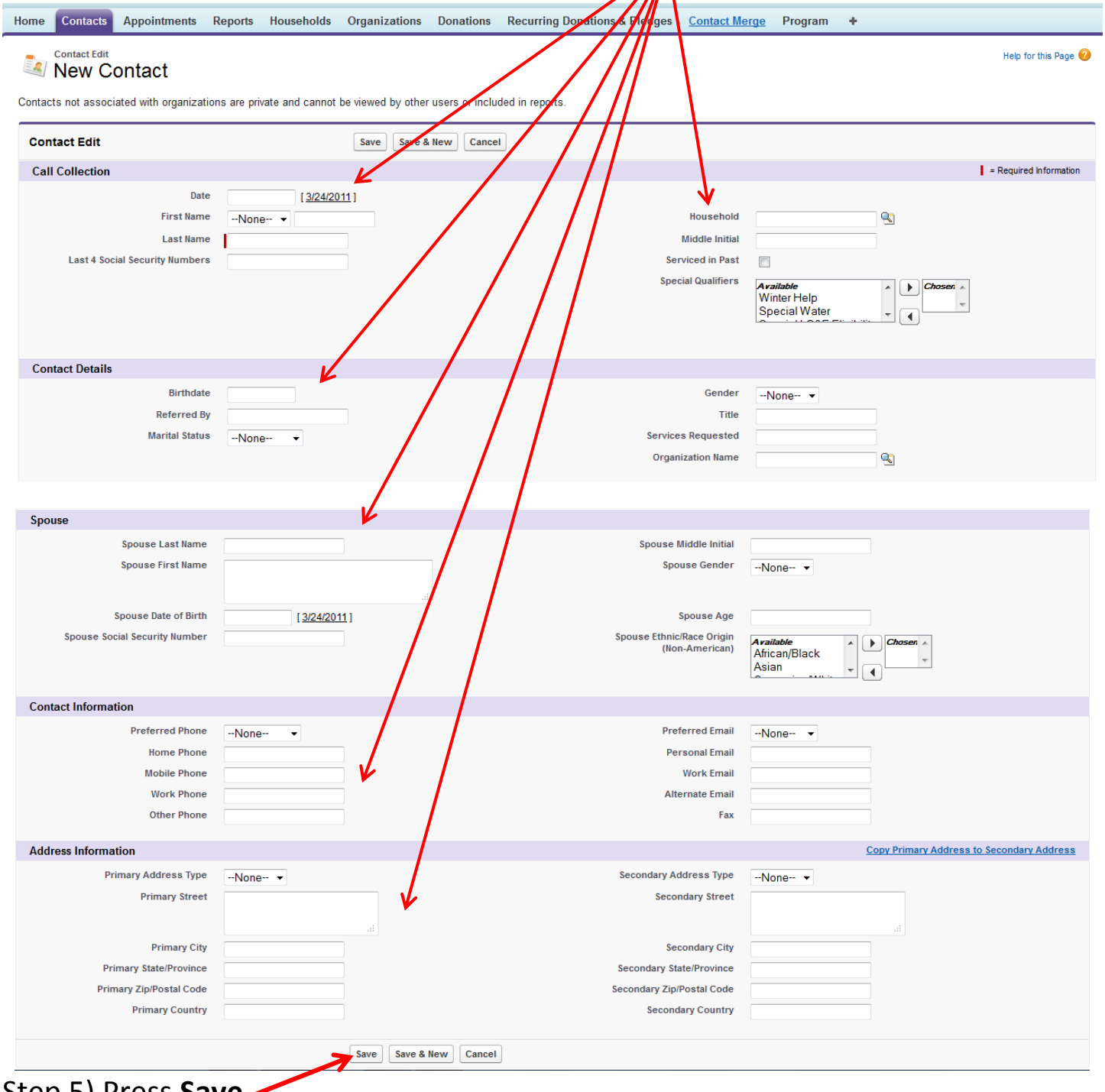
Step 2) Press **New**

Step 3) Choose **Client** or **Donor** from the dropdown



Log Name, Address, Schedule Appointment

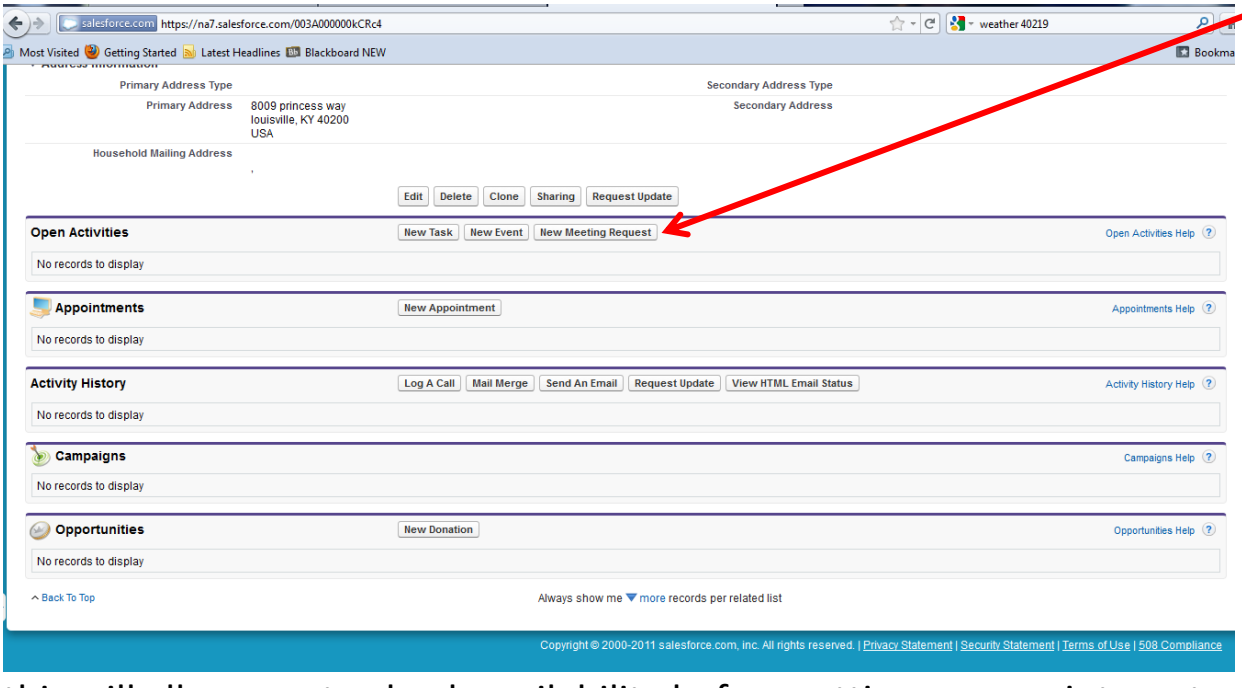
Step 4) Enter the client's information in the fields provided. Note: Fields with a red line at the beginning of the field indicate that the field is required. You may choose to enter non-required fields at any time (you can save the record, then edit it later for info that is not required at the time of the call)



Step 5) Press **Save**

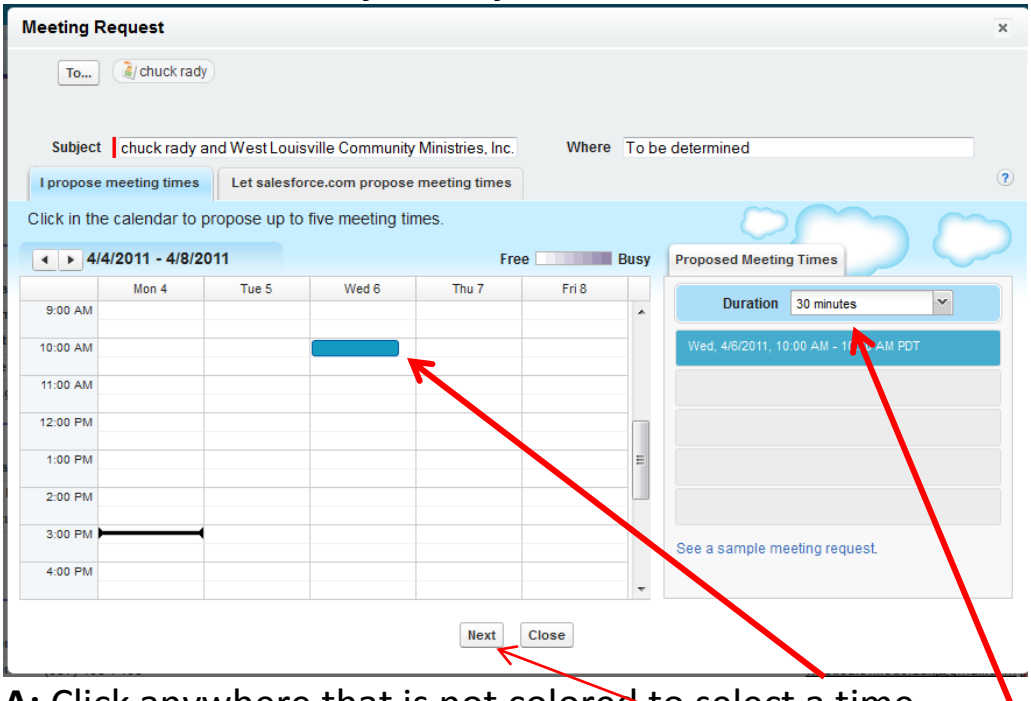
Log Name, Address, Schedule Appointment

Step 6) Scroll to the bottom of the record and select “new meeting request”



this will allow you to check availability before setting an appointment

On the next screen, you may use the visual calendar to see and select open times

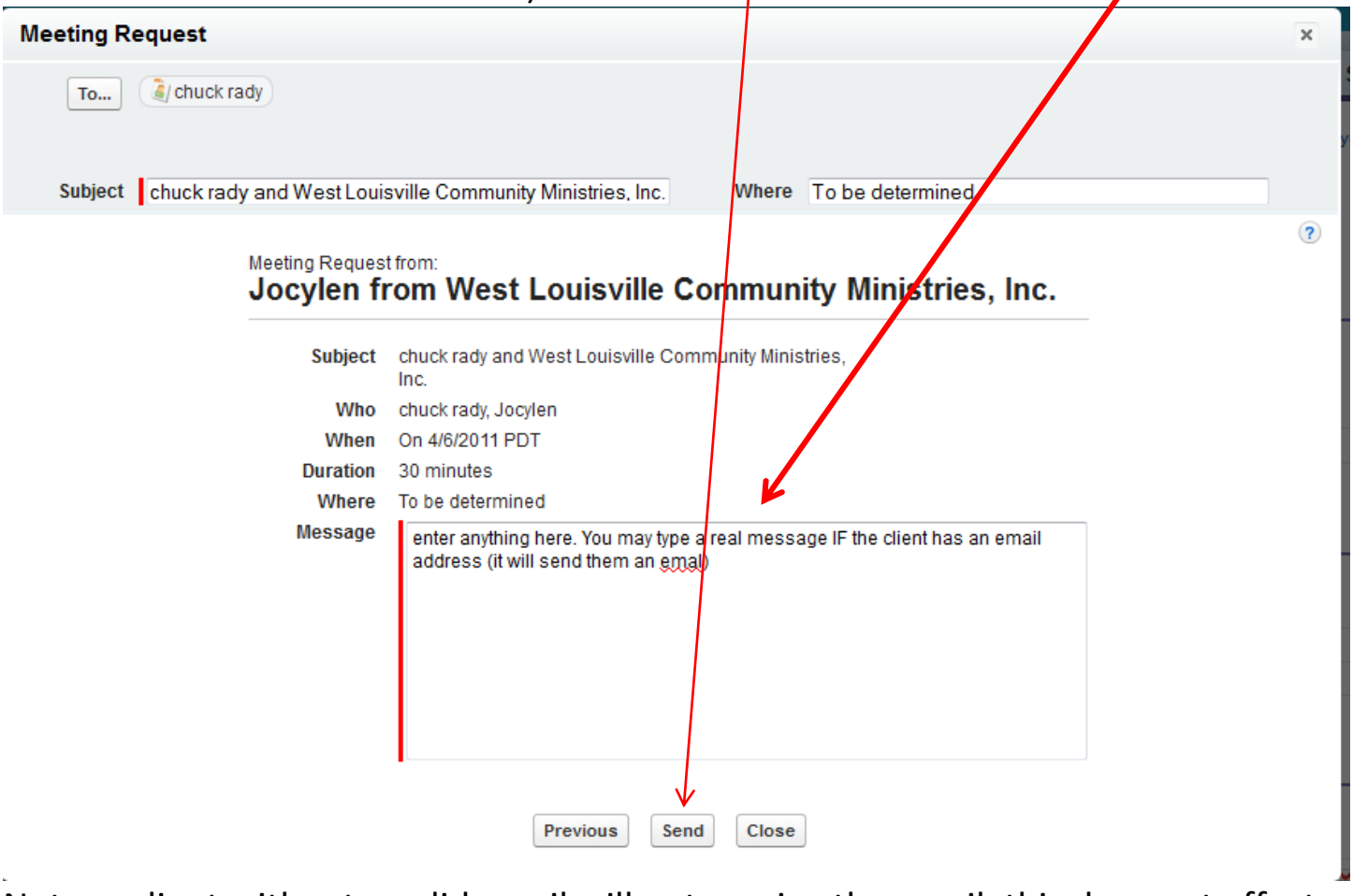


A: Click anywhere that is not colored to select a time

B: Set the time to 30 minutes

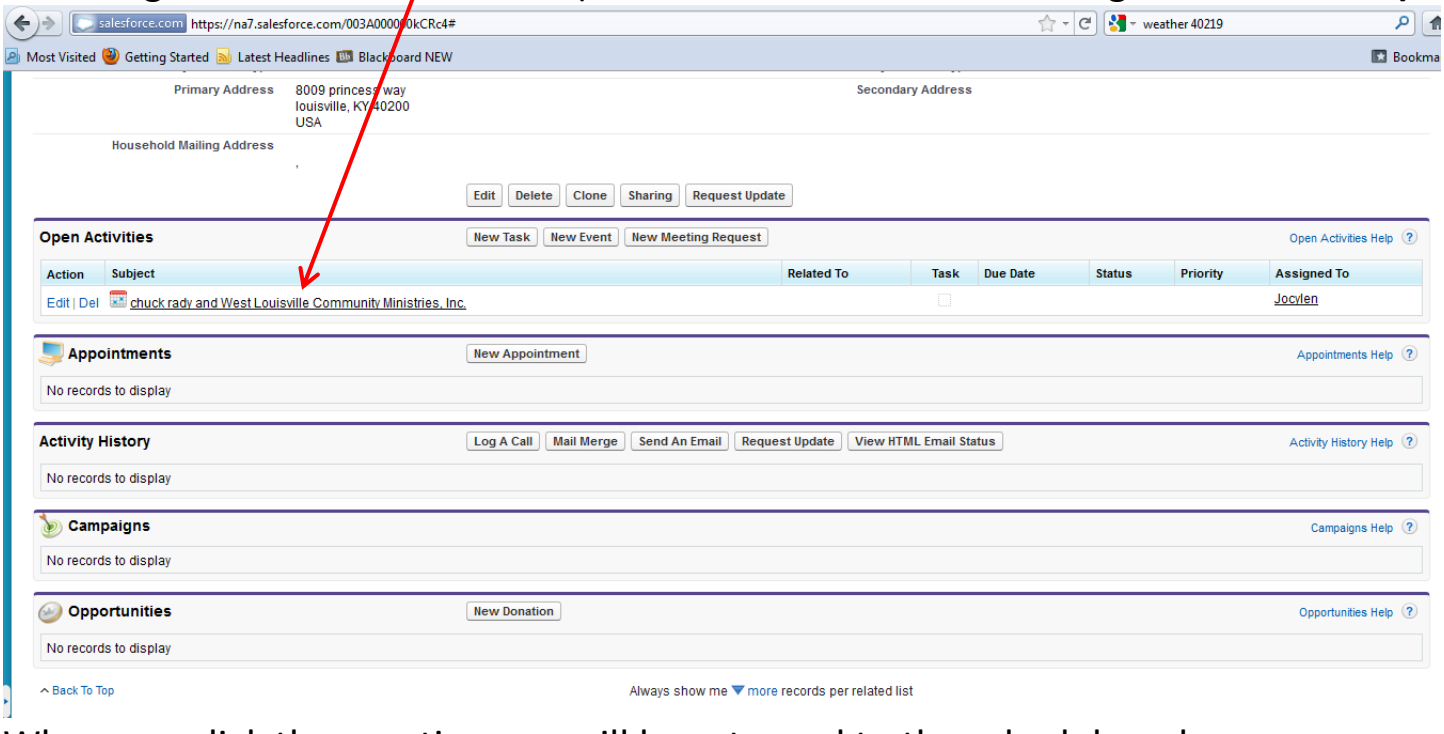
C: Press **Next**

On the next screen , you may type a message (or type any character if the client does not have an e-mail address) then click "**Send**"

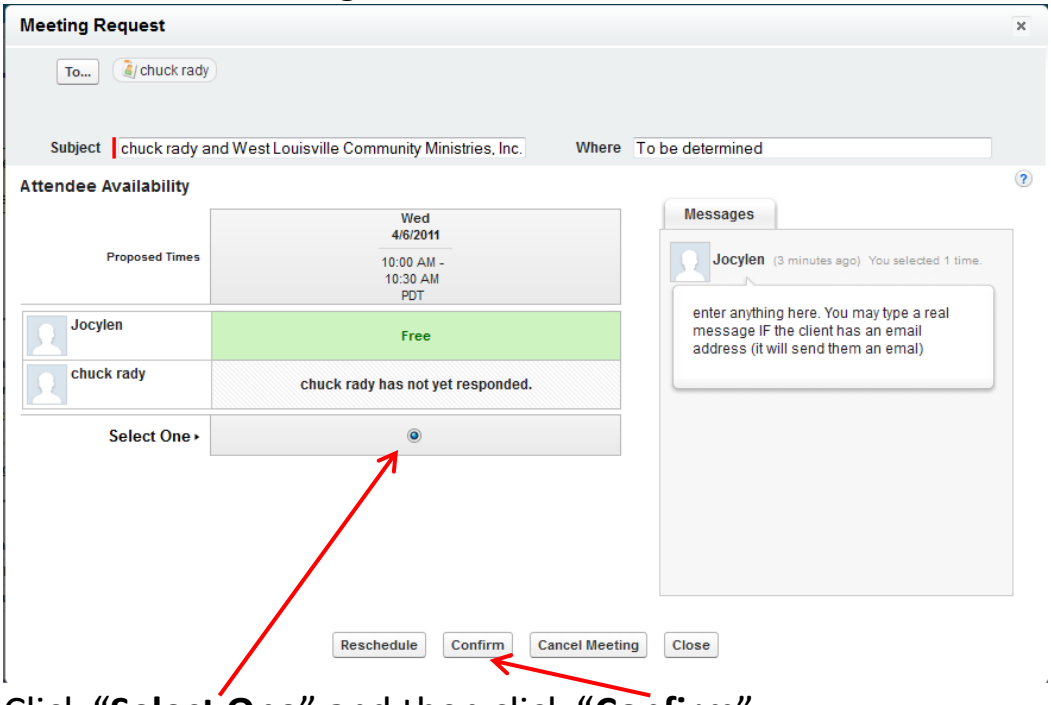


Note: a client without a valid email will not receive the email, this does not affect the scheduling within the system. If the client does have an email, he will receive the message. This can be used also to have clients confirm appointments via e-mail. Feel free to experiment with this.

You will now be returned to the client record. Once there, you will see the new "meeting" in Activities. Click this (to confirm it and add it to the general calendar**)**

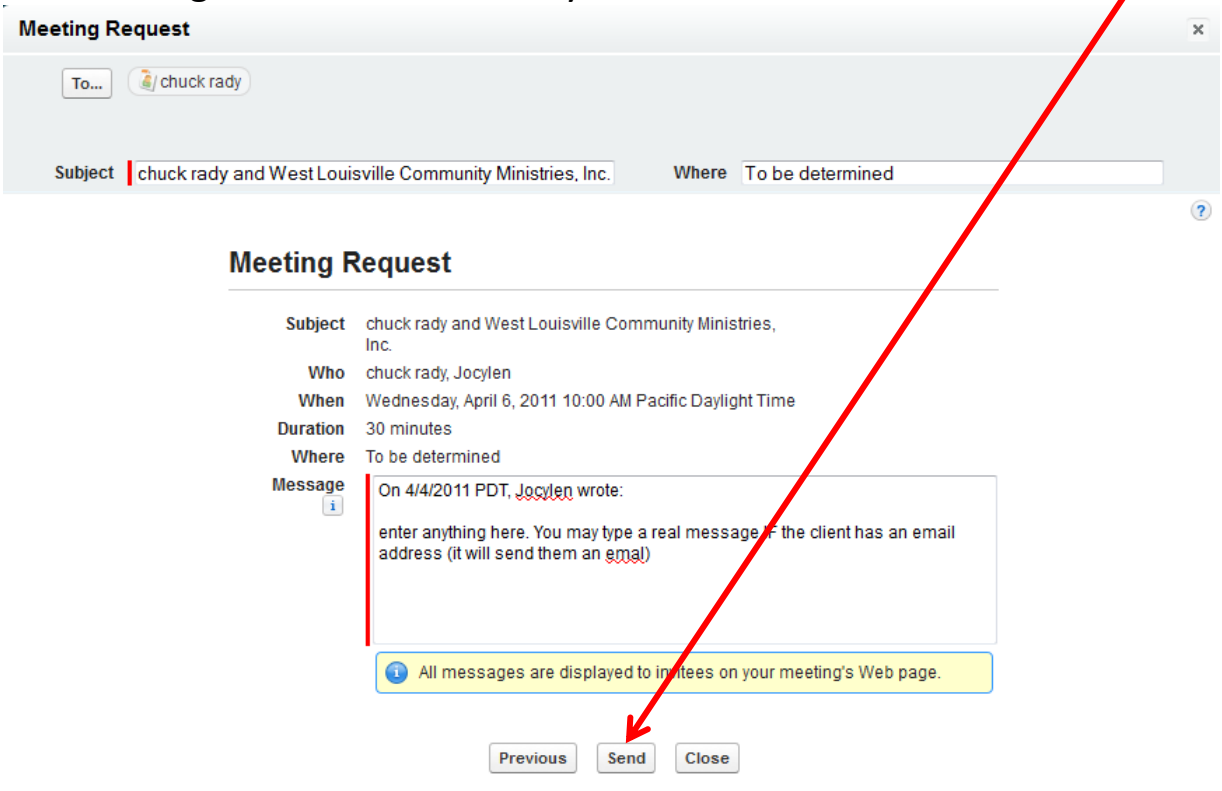


When you click the meeting you will be returned to the scheduler where you may confirm the meeting and block that time and date from being double-booked



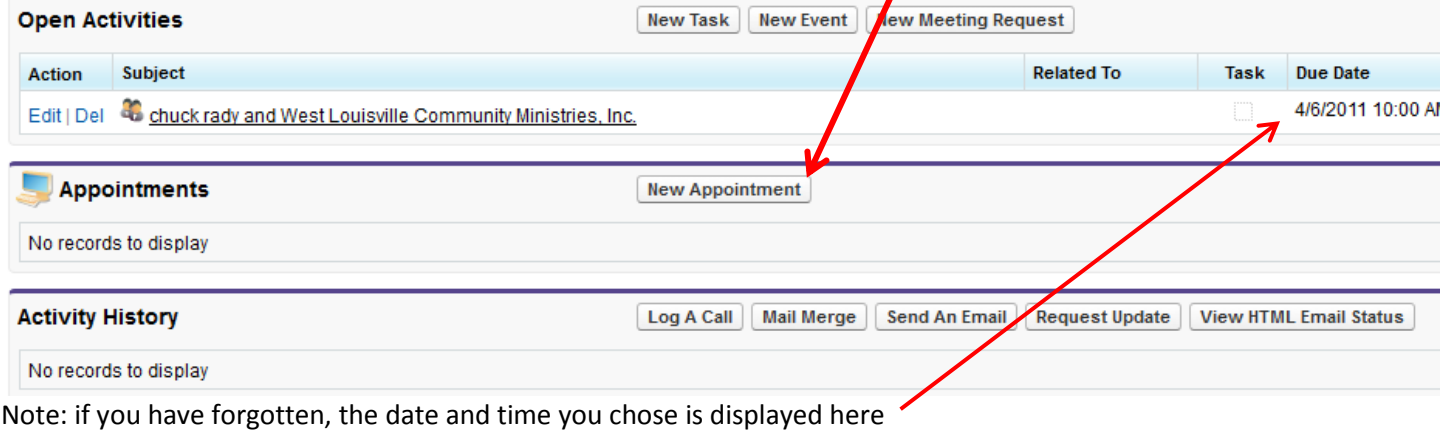
Click "**Select One**" and then click "**Confirm**"

The final screen of the scheduler will now appear, simply click “Send” to complete the booking and be automatically returned to the client record.



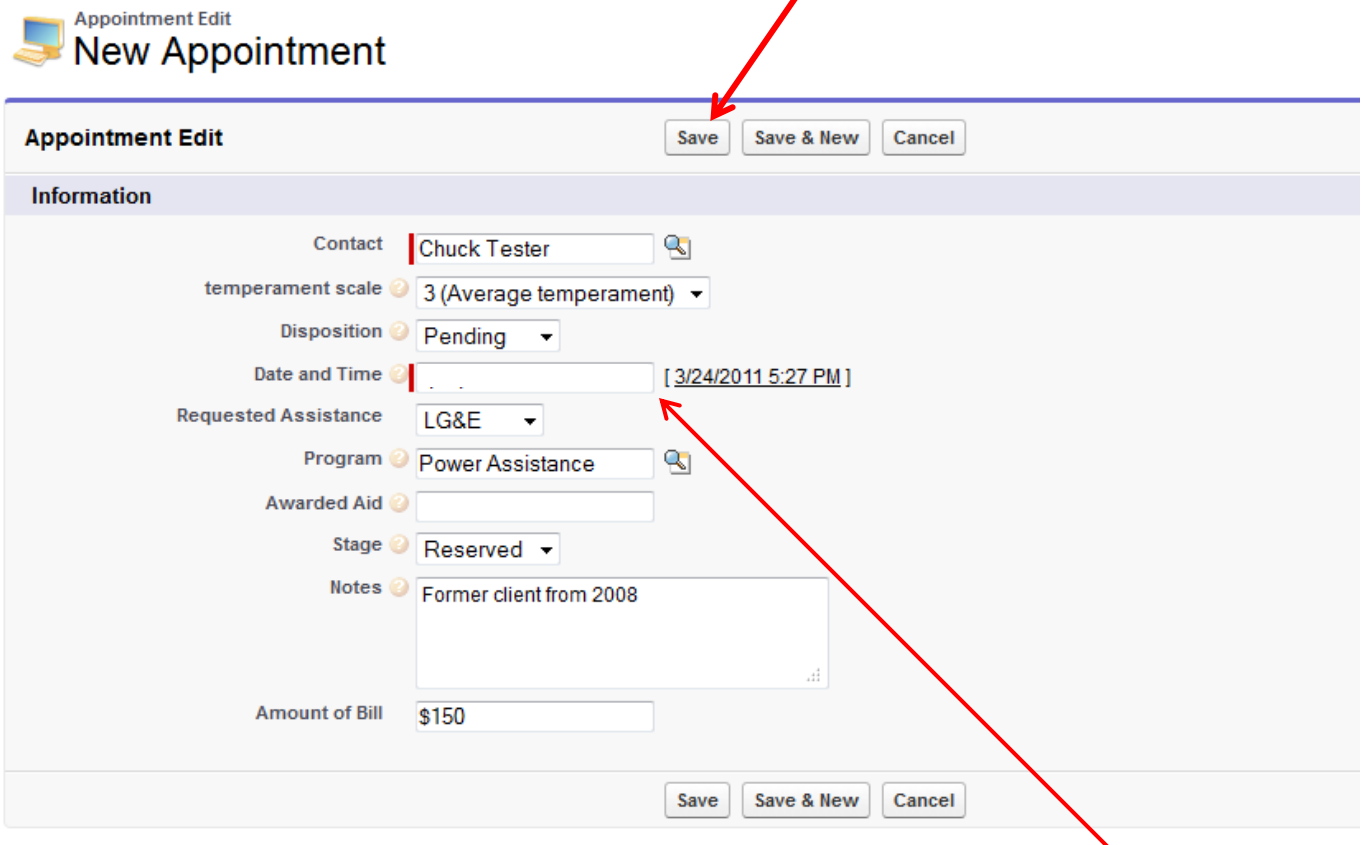
Create Appointment Date/Time

Step 7) At this time the appointment time will be booked within the scheduler (calendar) You may now set an official appointment and collect the data relevant to the appointment. Do this by clicking “**New Appointment”** *take note of the time and day you chose as you must re-enter it here (for reporting purposes)



Note: if you have forgotten, the date and time you chose is displayed here

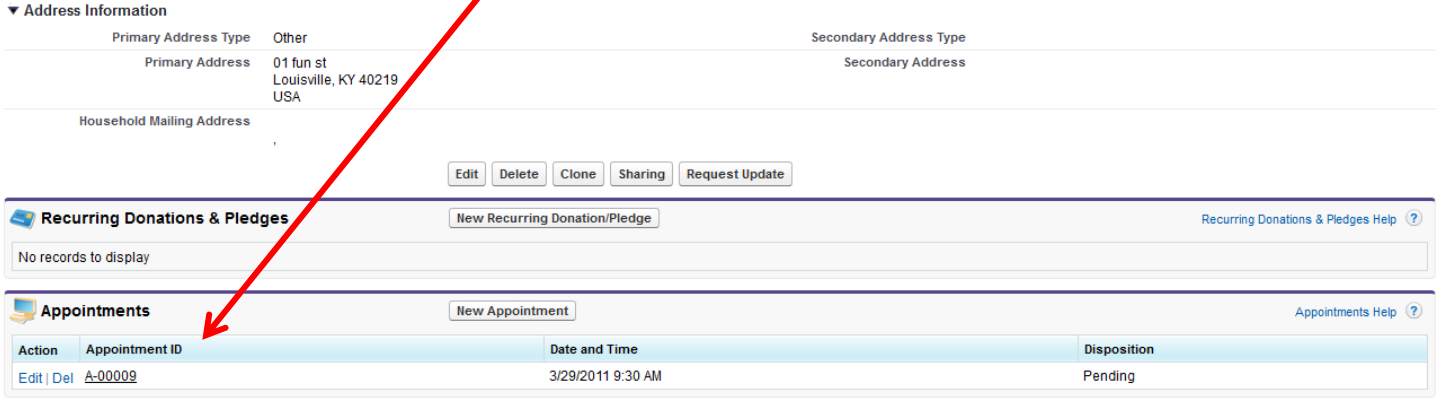
Step 8) Enter the appointment information and click **save**



***be sure to choose the same time and date here as the one you chose within the scheduler

****note: some lookup fields may be empty, in this case simply leave the field blank. We are changing some of the fields (program, etc)

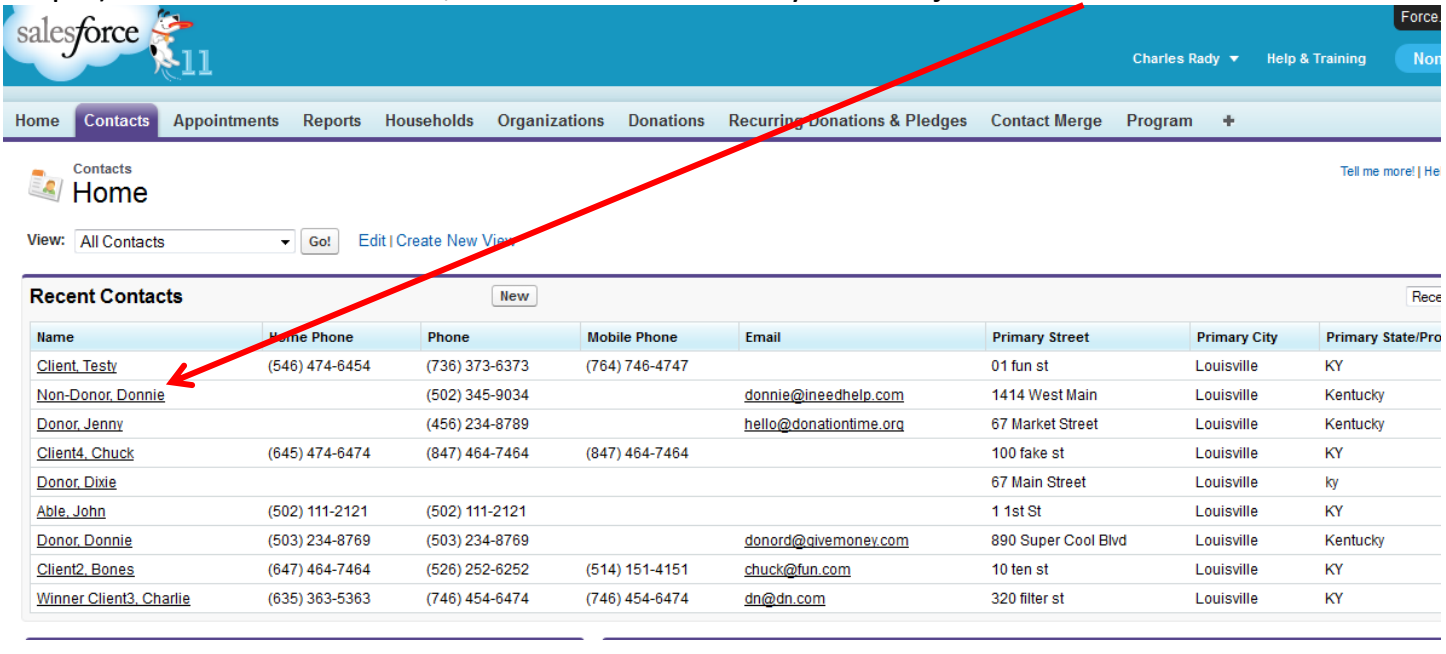
When you have completed this step, you will have successfully added both a client and their first appointment, click the contact tab again and you will see your new client in the list. To see all the clients click “go.” Select your client and you will see the appointment at the bottom of the page. Both the client and the appointment record can be edited.



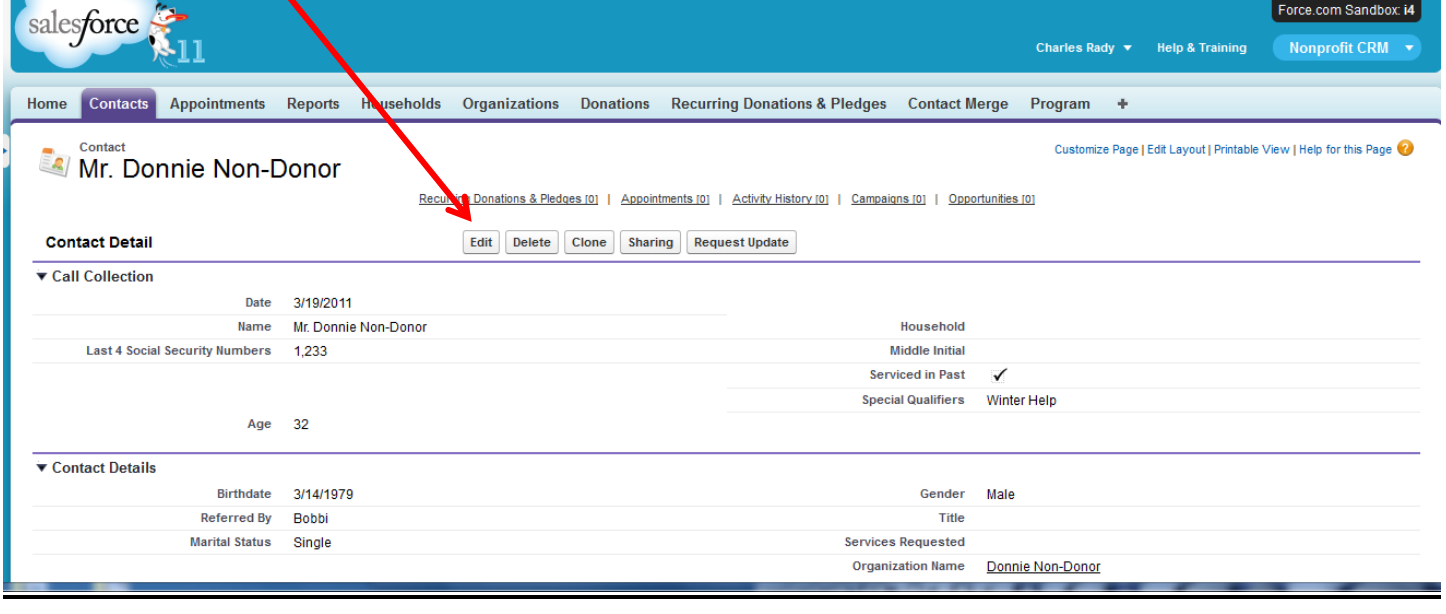
Meet Client

Record Spouse/Add'l Info

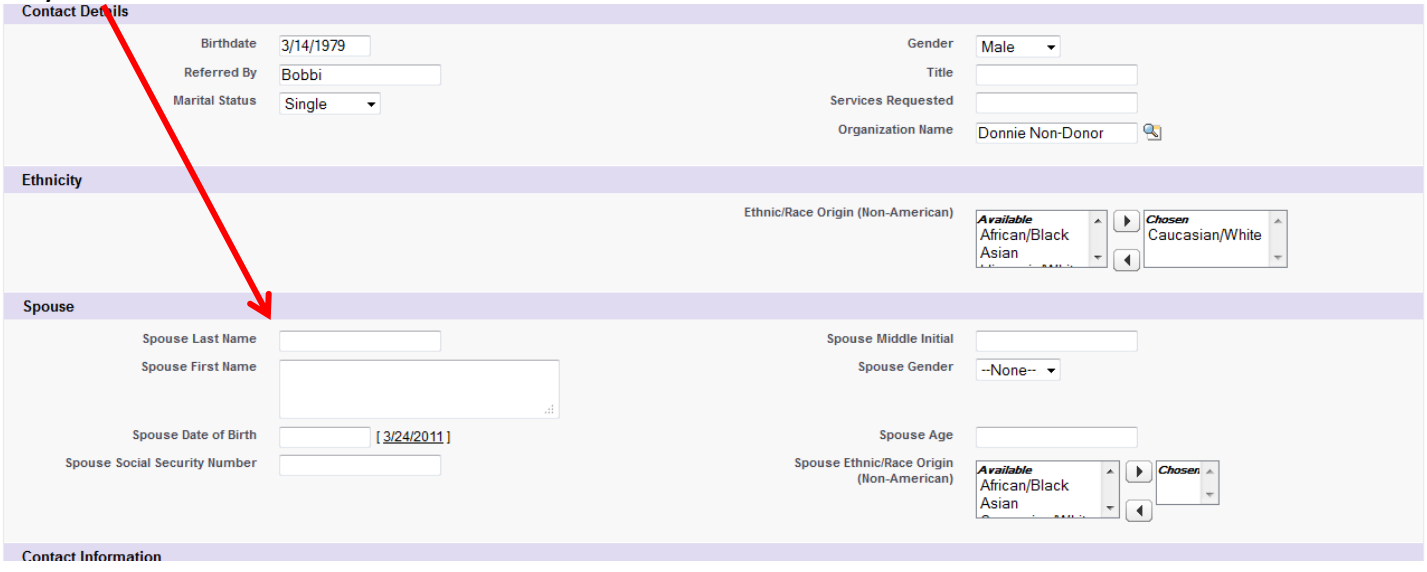
Step 9) Click the **contacts** tab, then select the client you have just created


Charles Rady
Help & Training
Home
Contacts
Appointments
Reports
Households
Organizations
Donations
Recurring Donations & Pledges
Contact Merge
Program
Contacts
Home
View:
All Contacts
Go!
Edit | Create New View
Recent Contacts
New
Name
Phone
Mobile Phone
Email
Primary Street
Primary City
Primary State/Pro
Client, Testy (546) 474-6454 (736) 373-6373 (764) 746-4747 01 fun st Louisville KY
Non-Donor, Donnie (502) 345-9034 donnie@ineedhelp.com 1414 West Main Louisville Kentucky
Donor, Jenny (456) 234-8789 hello@donationtime.org 67 Market Street Louisville Kentucky
Client4, Chuck (645) 474-6474 (847) 464-7464 (847) 464-7464 100 fake st Louisville KY
Donor, Dixie 67 Main Street Louisville ky
Able, John (502) 111-2121 (502) 111-2121 1 1st St Louisville KY
Donor, Donnie (503) 234-8769 (503) 234-8769 donord@givemoney.com 890 Super Cool Blvd Louisville Kentucky
Client2, Bones (647) 464-7464 (526) 252-6252 (514) 151-4151 chuck@fun.com 10 ten st Louisville KY
Winner Client3, Charlie (635) 363-5363 (746) 454-6474 (746) 454-6474 dn@dn.com 320 filter st Louisville KY

Step 10) Click **Edit**



Step 11)Scroll down to the spouse info (and any other additional info you wish to enter) You may edit the fields



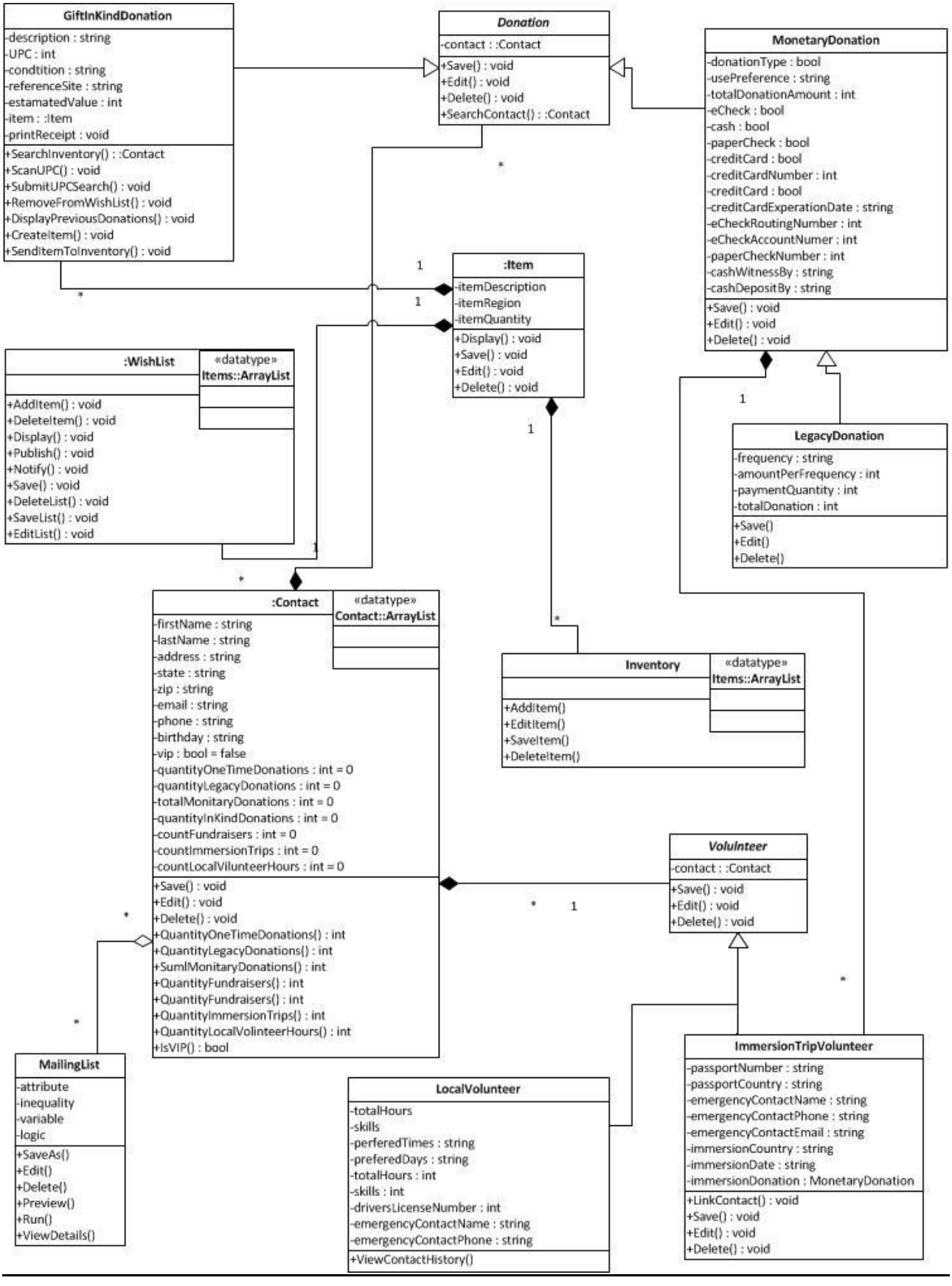
Step 12) Scroll to the bottom and click **save**

Note: When you log back in (or click "Home") you will see your upcoming meetings displayed.

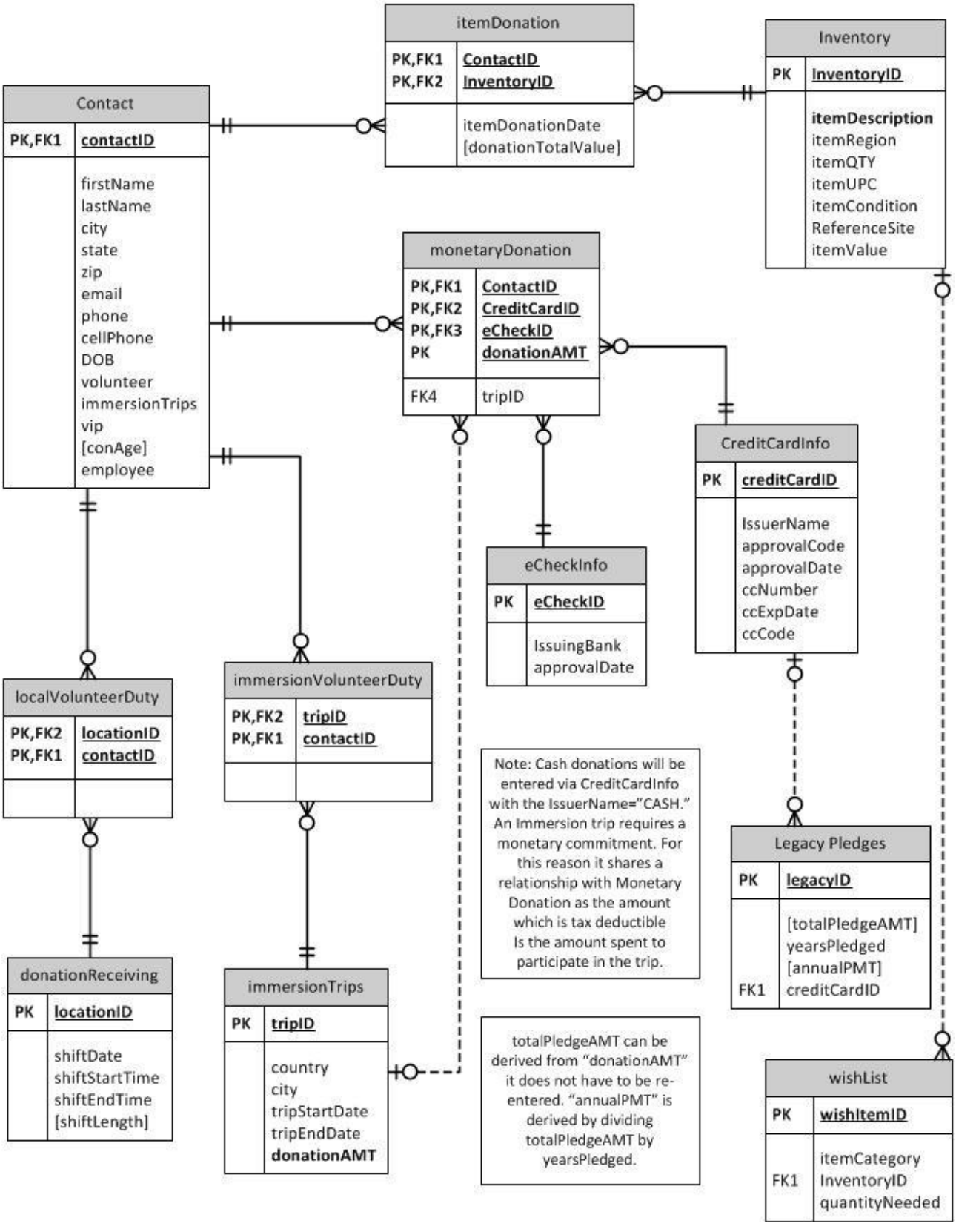
**The scheduling/Appointment workflow is still under development and will change with newer iterations. We are experimenting with a few different ways to do this.*

Step 13 is under development (income and expenses) you may experiment with anything you see within the SalesForce environment.

Iteration 5 Class Diagram

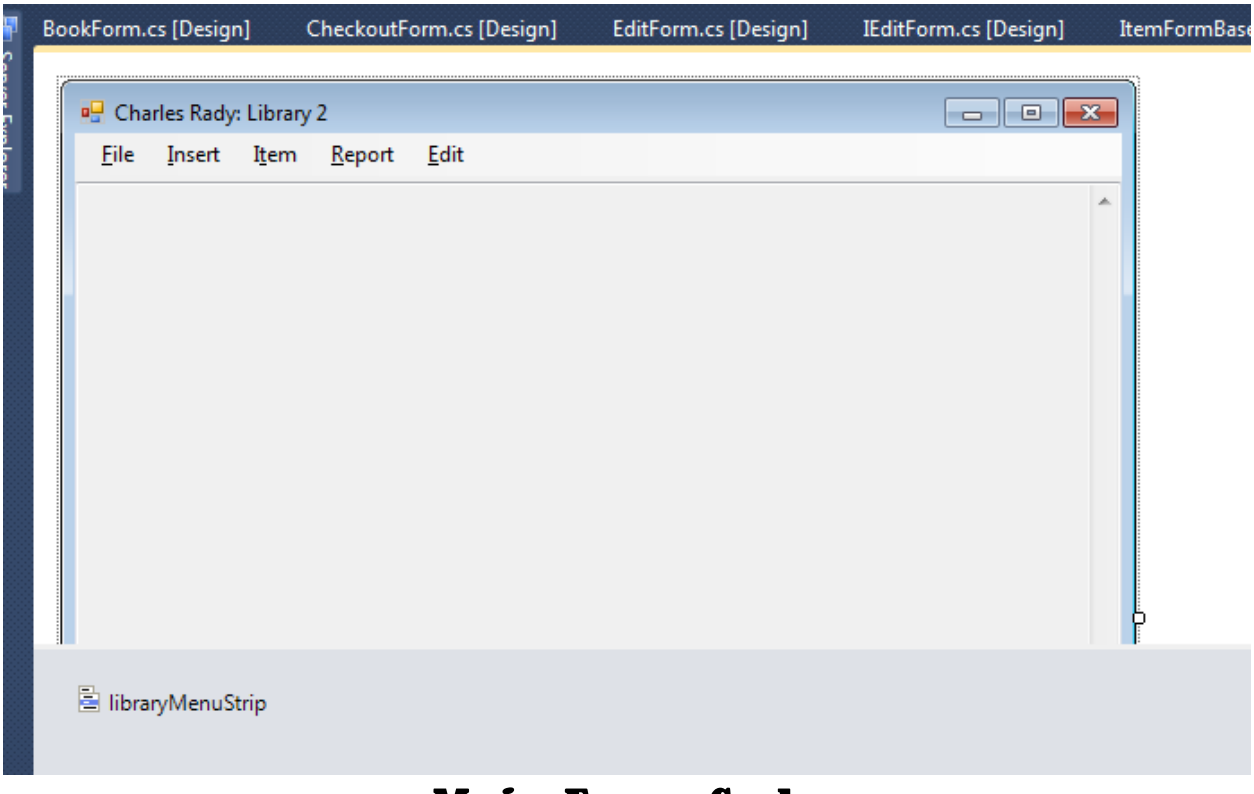


ERD from iteration 5



Library Program (version 2)

Library Code
FORMS
Main Form



Main Form Code

```
// Program 3
// CIS 200-01
// Spring 2011
// Due: 4/6/2011
// By: Charles Rady

// File: Prog2Form.cs
// This class creates the main GUI for Program 2. It provides a
// File menu with About and Exit items, an Insert menu with Patron and
// Book items, an Item menu with Check Out and Return items, and a
// Report menu with Patron List, Item List, and Checked Out Items items.
// Extra Credit - Check Out and Return only show relevant items
//it now also has the capability of saving and opening files as well as editing patrons and items

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using System.IO;
using System.Runtime.Serialization;
using System.Runtime.Serialization.Formatters.Binary;
```

```
namespace LibraryItems
{
    [Serializable]
    public partial class Prog2Form : Form
    {
        private Library lib;                 // The library
        private List<LibraryItem> items;     // List of library items
        private List<LibraryPatron> patrons; // List of patrons
        private FileStream output; // stream for saving to a file
        private FileStream input; // stream for writing to the library from the file
        private BinaryFormatter formatter = new BinaryFormatter();// create binary formatter for saving
        private BinaryFormatter reader = new BinaryFormatter(); // create binary formatter opening

        // Precondition:  None
        // Postcondition: The form's GUI is prepared for display. A few test items and patrons
        //                are added to the library
        public Prog2Form()
        {
            InitializeComponent();

            lib = new Library(); // Create the library


            //magic items and patrons removed.


            items = lib.GetItemsList();
            patrons = lib.GetPatronsList();

        }

        // Precondition:  File, About menu item activated
        // Postcondition: Information about author displayed in dialog box
        private void aboutToolStripMenuItem_Click(object sender, EventArgs e)
        {
            MessageBox.Show(String.Format("Program 3 - Charles Rady{0}" +
                "CIS 200-01{0}Spring 2011", System.Environment.NewLine), "About Program 3");
        }

        // Precondition:  File, Exit menu item activated
        // Postcondition: The application is exited
        private void exitToolStripMenuItem_Click(object sender, EventArgs e)
        {
            Application.Exit();
        }

        // Precondition:  Report, Patron List menu item activated
        // Postcondition: The list of patrons is displayed in the reportTxt
        //                text box
        private void patronListToolStripMenuItem_Click(object sender, EventArgs e)
        {
            StringBuilder result = new StringBuilder(); // Holds text as report being built
            // StringBuilder more efficient than String

            result.Append(String.Format("Patron List - {0} patrons", patrons.Count()));
            result.Append(System.Environment.NewLine); // Remember, \n doesn't always work in GUIs
            result.Append(System.Environment.NewLine);

            foreach (LibraryPatron p in patrons)
            {
                result.Append(p.ToString());
                result.Append(System.Environment.NewLine);
                result.Append(System.Environment.NewLine);
```

```
    }

    reportTxt.Text = result.ToString();

    // Put cursor at start of report
    reportTxt.Focus();
    reportTxt.SelectionStart = 0;
    reportTxt.SelectionLength = 0;
}

// Precondition:  Report, Item List menu item activated
// Postcondition: The list of items is displayed in the reportTxt
//                text box
private void itemListToolStripMenuItem_Click(object sender, EventArgs e)
{
    StringBuilder result = new StringBuilder(); // Holds text as report being built
    // StringBuilder more efficient than String

    result.Append(String.Format("Item List - {0} items", items.Count()));
    result.Append(System.Environment.NewLine); // Remember, \n doesn't always work in GUIs
    result.Append(System.Environment.NewLine);

    foreach (LibraryItem item in items)
    {
        result.Append(item.ToString());
        result.Append(System.Environment.NewLine);
        result.Append(System.Environment.NewLine);
    }

    reportTxt.Text = result.ToString();

    // Put cursor at start of report
    reportTxt.Focus();
    reportTxt.SelectionStart = 0;
    reportTxt.SelectionLength = 0;
}

// Precondition:  Report, Checked Out Items menu item activated
// Postcondition: The list of checked out items is displayed in the
//                reportTxt text box
private void checkedOutItemsToolStripMenuItem_Click(object sender, EventArgs e)
{
    StringBuilder result = new StringBuilder(); // Holds text as report being built
    // StringBuilder more efficient than String
    // LINQ: selects checked out items
    var checkedOutItems =
        from item in items
        where item.IsCheckedOut()
        select item;

    result.Append(String.Format("Checked Out Items - {0} items", checkedOutItems.Count()));
    result.Append(System.Environment.NewLine); // Remember, \n doesn't always work in GUIs
    result.Append(System.Environment.NewLine);

    foreach (LibraryItem item in checkedOutItems)
    {
        result.Append(item.ToString());
        result.Append(System.Environment.NewLine);
        result.Append(System.Environment.NewLine);
    }

    reportTxt.Text = result.ToString();
```

```
            // Put cursor at start of report
            reportTxt.Focus();
            reportTxt.SelectionStart = 0;
            reportTxt.SelectionLength = 0;
        }

        // Precondition:  Insert, Patron menu item activated
        // Postcondition: The Patron dialog box is displayed. If data entered
        //                are OK, a LibraryPatron is created and added to the library
        private void patronToolStripMenuItem_Click(object sender, EventArgs e)
        {
            PatronForm patronForm = new PatronForm(); // The patron dialog box form

            DialogResult result = patronForm.ShowDialog(); // Show form as dialog and store result

            if (result == DialogResult.OK) // Only add if OK
            {
                // Use form's properties to get patron info to send to library
                //lib.AddPatron(patronForm.PatronName, patronForm.PatronID);
                LibraryPatron newPatron = new LibraryPatron(patronForm.PatronName,
patronForm.PatronID);
                patrons.Add(newPatron);
            }

            patronForm.Dispose(); // Good .NET practice - will get garbage collected anyway
        }

        // Precondition:  Insert, Book menu item activated
        // Postcondition: The Book dialog box is displayed. If data entered
        //                are OK, a LibraryBook is created and added to the library
        private void bookToolStripMenuItem_Click(object sender, EventArgs e)
        {
            BookForm bookForm = new BookForm(); // The book dialog box form

            DialogResult result = bookForm.ShowDialog(); // Show form as dialog and store result

            if (result == DialogResult.OK) // Only add if OK
            {
                try
                {
                    // Use form's properties to get book info to send to library
                    lib.AddLibraryBook(bookForm.ItemTitle, bookForm.ItemPublisher,
int.Parse(bookForm.ItemCopyrightYear),
                        int.Parse(bookForm.ItemLoanPeriod), bookForm.ItemCallNumber,
bookForm.BookAuthor);
                }

                catch (FormatException) // This should never happen if form validation works!
                {
                    MessageBox.Show("Problem with Book Validation!", "Validation Error");
                }
            }

            bookForm.Dispose(); // Good .NET practice - will get garbage collected anyway
        }

        // Precondition:  Item, Check Out menu item activated
        // Postcondition: The Checkout dialog box is displayed. If data entered
        //                are OK, an item is checked out from the library by a patron
        private void checkOutToolStripMenuItem_Click(object sender, EventArgs e)
        {
            // Extra Credit - Only display items that aren't already checked out
```

```
            List<LibraryItem> notCheckedOutList; // List of items not checked out
            List<int> notCheckedOutIndices;      // List of index values of items not checked out

            notCheckedOutList = new List<LibraryItem>();
            notCheckedOutIndices = new List<int>();

            for (int i = 0; i < items.Count(); ++i)
                if (!items[i].IsCheckedOut()) // Not checked out
                {
                    notCheckedOutList.Add(items[i]);
                    notCheckedOutIndices.Add(i);
                }

            if ((notCheckedOutList.Count() == 0) || (patrons.Count() == 0)) // Must have items and
patrons
                MessageBox.Show("Must have items and patrons to check out!", "Check Out Error");
            else
            {
                CheckoutForm checkoutForm = new CheckoutForm(notCheckedOutList, patrons); // The check
out dialog box form

                DialogResult result = checkoutForm.ShowDialog(); // Show form as dialog and store
result

                if (result == DialogResult.OK) // Only add if OK
                {
                    try
                    {
                        int itemIndex; // Index of item from full list of items

                        itemIndex = notCheckedOutIndices[checkoutForm.ItemIndex]; // Look up index from
full list
                        lib.CheckOut(itemIndex, checkoutForm.PatronIndex);
                    }
                    catch (ArgumentOutOfRangeException) // This should never happen
                    {
                        MessageBox.Show("Problem with Check Out Index!", "Check Out Error");
                    }
                }
                checkoutForm.Dispose(); // Good .NET practice - will get garbage collected anyway
            }
        }

        // Precondition:  Item, Return menu item activated
        // Postcondition: The Return dialog box is displayed. If data entered
        //                are OK, an item is returned to the library
        private void returnToolStripMenuItem_Click(object sender, EventArgs e)
        {
            // Extra Credit - Only display items that are already checked out

            List<LibraryItem> checkedOutList; // List of items checked out
            List<int> checkedOutIndices;      // List of index values of items checked out

            checkedOutList = new List<LibraryItem>();
            checkedOutIndices = new List<int>();

            for (int i = 0; i < items.Count(); ++i)
                if (items[i].IsCheckedOut()) // Checked out
                {
                    checkedOutList.Add(items[i]);
                    checkedOutIndices.Add(i);
                }
```

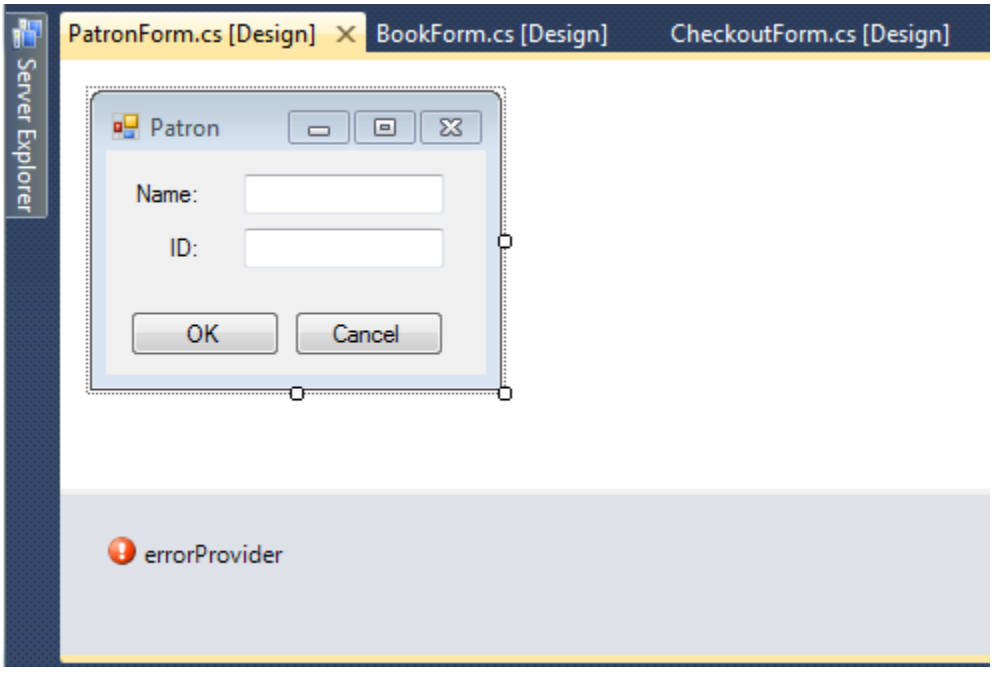
```
            if ((checkedOutList.Count() == 0)) // Must have checked out items
                MessageBox.Show("Must have checked out items to return!", "Return Error");
            else
            {
                ReturnForm returnForm = new ReturnForm(checkedOutList); // The return dialog box form

                DialogResult result = returnForm.ShowDialog(); // Show form as dialog and store result

                if (result == DialogResult.OK) // Only add if OK
                {
                    try
                    {
                        int itemIndex; // Index of item from full list of items

                        itemIndex = checkedOutIndices[returnForm.ItemIndex]; // Look up index from full
list
                        lib.ReturnToShelf(itemIndex);
                    }
                    catch (ArgumentOutOfRangeException) // This should never happen
                    {
                        MessageBox.Show("Problem with Return Index!", "Return Error");
                    }
                }
                returnForm.Dispose(); // Good .NET practice - will get garbage collected anyway
            }
        }
        // Precondition:  file, save item has been clicked
        // Postcondition: The library has been saved to a file specified by the user
        private void saveToolStripMenuItem_Click(object sender, EventArgs e)
        {
            // create and show dialog box for saving file
            DialogResult result;//local variable to store result
            string fileName; // string to hold saved file's name

            using (SaveFileDialog fileSaver = new SaveFileDialog())//allow use of SaveFileDialog
            {
                fileSaver.CheckFileExists = false;// check to see if the file already exists
                result = fileSaver.ShowDialog();  // show the dialog box
                fileName = fileSaver.FileName; // get file name
            } // end using

            if (result == DialogResult.OK)  // check for OK click
            {
                if (fileName == string.Empty) //check for empty string
                    MessageBox.Show("Invaild File Name", "Error",
                        MessageBoxButtons.OK, MessageBoxIcon.Error);//display error
                else
                {   // save file stream if user entered valid name
                    try
                    {
                        output = new FileStream(fileName, FileMode.OpenOrCreate,
FileAccess.Write);//set output stream to write to file
                        formatter.Serialize(output, lib);//write the library object to the file
                        output.Close();//close the stream
                    }
                    catch (IOException)
                    {

                        MessageBox.Show("Error opening file.", "Error",
                           MessageBoxButtons.OK, MessageBoxIcon.Error);// show error
                    } // end catch

                }//end try
```

```
            }//end if
        }//end save method

        // Precondition:  file, open has been clicked
        // Postcondition: the selected file has been opened and used to populate the library
        private void openToolStripMenuItem_Click(object sender, EventArgs e)
        {
            // create and show dialog box
            DialogResult result; // OK
            Library lib = new Library();//new library
            string fileName;//local variable to hold the filename
            using (OpenFileDialog fileChooser = new OpenFileDialog())
            {
                result = fileChooser.ShowDialog();//show the dialog
                fileName = fileChooser.FileName; // get specified file name
            } // end using

            if (result == DialogResult.OK) // user clicks ok
            {
                lib = null; // delete old library
                try
                {
                    // open file stream
                    input = new FileStream(fileName, FileMode.Open, FileAccess.Read);//input the stream
of data from the file
                    lib = (Library)reader.Deserialize(input); // deserialize stream
                    items = lib.items;//replace the list with the list from the opened file
                    patrons = lib.patrons;//replace the list with the list from the opened file

                    input.Close(); // close file stream
                }
                catch (SerializationException)//catch exception thrown by serialization error
                {
                    MessageBox.Show("Invalid File Name 5 ", "Error",
                                    MessageBoxButtons.OK, MessageBoxIcon.Error);//show error in message
box
                } // end catch
            }//end if
        }// end open method


        // Precondition:  edit, patron has been clicked
        // Postcondition: the patron selected by the user is updated with the new data
        private void patronToolStripMenuItem1_Click(object sender, EventArgs e)

            {
                EditForm patronSelect = new EditForm(patrons); // create instance of patronSelect,
populate with list
                DialogResult selectResult = patronSelect.ShowDialog(); //use dialog to display form
                if (selectResult == DialogResult.OK)
                {
                    PatronForm patronForm = new PatronForm(); // make a new patron form
                    //fill the fields with the data from the selected patron
                    patronForm.PatronName = patrons[patronSelect.PatronIndex].PatronName;//name field
                    patronForm.PatronID = patrons[patronSelect.PatronIndex].PatronID;//ID field
                    DialogResult result = patronForm.ShowDialog(); // Show form
                    if (result == DialogResult.OK)//check for OK click



                    {
                        try
                        {
```

```
                            patrons.RemoveAt(patronSelect.PatronIndex);//remove original patron
                            LibraryPatron newPatron = new LibraryPatron(patronForm.PatronName,
patronForm.PatronID);//make a new patron to use with AddPatron method
                            patrons.Add(newPatron);//add the new data
                        }
                        catch (FormatException)//catch format exception
                        {
                            MessageBox.Show("Format incorrect", "Error", MessageBoxButtons.OK,
MessageBoxIcon.Error);//display error
                        }



                        patronSelect.Dispose(); // collect it and toss it
                    }//end if

                }//end if
            }//end patron edit method


        // Precondition:  edit, item has been clicked
        // Postcondition: the item selected by the user has been updated with new data
        private void itemToolStripMenuItem1_Click(object sender, EventArgs e)

            {

                IEditForm itemSelect = new IEditForm(items); // create instance of the item edit form,
populate with list
                DialogResult selectResult = itemSelect.ShowDialog(); //display form

                if (selectResult == DialogResult.OK)
                {

                    BookForm bookForm = new BookForm(); // the add book form

                    //Enter the chosen item's data into the appropriate fields for editing
                    try//exception handling (in case of empty choice)
                    {
                        bookForm.ItemTitle = items[itemSelect.itemIndex].Title;//title field
                        bookForm.ItemPublisher = items[itemSelect.itemIndex].Publisher;//publisher
field
                        bookForm.ItemCopyrightYear =
Convert.ToString(items[itemSelect.itemIndex].CopyrightYear);//copyright field
                        bookForm.ItemLoanPeriod =
Convert.ToString(items[itemSelect.itemIndex].LoanPeriod);//loan field
                        bookForm.ItemCallNumber = items[itemSelect.itemIndex].CallNumber;//callnumber
field
                        bookForm.BookAuthor =
((LibraryBook)items[itemSelect.itemIndex]).Author;//downcast the author field
                    }
                    //catch exception created if nothing is chosen
                    catch { MessageBox.Show("Error, you have not chosen an item. You may now create a
new item, or cancel and try again.","ERROR"); }

                        DialogResult result = bookForm.ShowDialog(); // Show form
                        if (result == DialogResult.OK)//see if user clicked OK
                        {
                            try
                            {


                                items.RemoveAt(itemSelect.itemIndex);//remove original data
```

```
                                LibraryBook newItem = new LibraryBook(bookForm.ItemTitle,
bookForm.ItemPublisher, int.Parse(bookForm.ItemCopyrightYear), int.Parse(bookForm.ItemLoanPeriod),
bookForm.ItemCallNumber, bookForm.BookAuthor);//make a new book with AddLibraryBook method
                                items.Add(newItem);//add updated data

                            }
                            catch (FormatException)
                            {
                                MessageBox.Show("Format Error", "Error", MessageBoxButtons.OK,
MessageBoxIcon.Error); //display error
                            }



                            itemSelect.Dispose(); // collect it and toss it
                        }//end if

                    }//end if
                }//end method
            }
        }
```

Patron Form



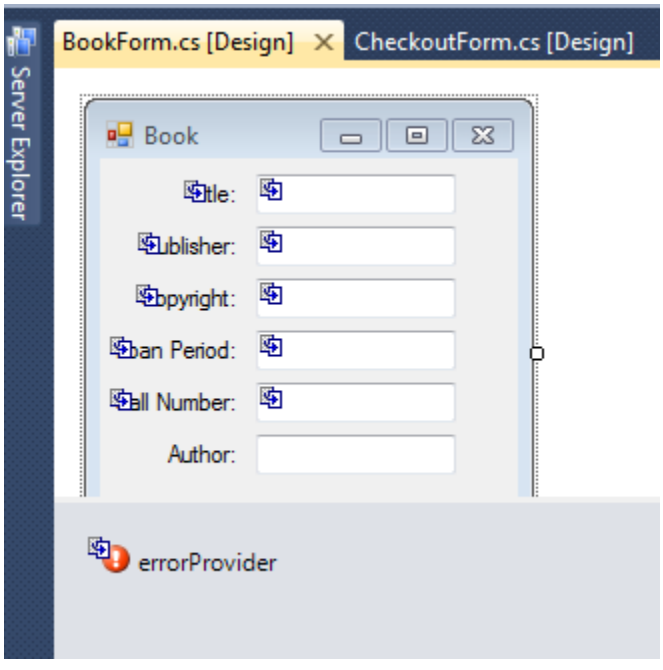
Patron Form Code

```
// Program 2
// CIS 200-01
// Spring 2011
// Due: 4/6/2011
//Charles Rady

// File: PatronForm.cs
// This class creates the Patron dialog box form GUI. It performs validation
// and provides String properties for each field.

using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;

namespace LibraryItems
{
    public partial class PatronForm : Form
    {
        // Precondition:  None
        // Postcondition: The form's GUI is prepared for display.
        public PatronForm()
        {
            InitializeComponent();
        }

        public String PatronName
        {
            // Precondition:  None
            // Postcondition: The text of form's name field has been returned
            get
            {
```

```
            return patronNameTxt.Text;
        }

        // Precondition:  None
        // Postcondition: The text of form's name field has been set to the specified value
        set
        {
            patronNameTxt.Text = value;
        }
    }

    public String PatronID
    {
        // Precondition:  None
        // Postcondition: The text of form's ID field has been returned
        get
        {
            return patronIdTxt.Text;
        }

        // Precondition:  None
        // Postcondition: The text of form's ID field has been set to the specified value
        set
        {
            patronIdTxt.Text = value;
        }
    }

    // Precondition:  Focus is shifting from patronNameTxt
    // Postcondition: If text is invalid, focus remains and error provider
    //                highlights the field
    private void patronNameTxt_Validating(object sender, CancelEventArgs e)
    {
        if (patronNameTxt.TextLength == 0) // Empty field
        {
            e.Cancel = true;
            errorProvider.SetError(patronNameTxt, "Must provide Name");
        }
    }

    // Precondition:  Validating of patronNameTxt not cancelled, so data OK
    // Postcondition: Error provider cleared and focus allowed to change
    private void patronNameTxt_Validated(object sender, EventArgs e)
    {
        errorProvider.SetError(patronNameTxt, "");
    }

    // Precondition:  Focus is shifting from patronIdTxt
    // Postcondition: If text is invalid, focus remains and error provider
    //                highlights the field
    private void patronIdTxt_Validating(object sender, CancelEventArgs e)
    {
        if (patronIdTxt.TextLength == 0) // Empty field
        {
            e.Cancel = true;
            errorProvider.SetError(patronIdTxt, "Must provide ID");
        }
    }

    // Precondition:  Validating of patronIdTxt not cancelled, so data OK
    // Postcondition: Error provider cleared and focus allowed to change
    private void patronIdTxt_Validated(object sender, EventArgs e)
    {
```

```
            errorProvider.SetError(patronIdTxt, "");
        }

        // Precondition:  User pressed on cancelBtn
        // Postcondition: Form closes and sends Cancel result
        private void cancelBtn_MouseDown(object sender, MouseEventArgs e)
        {
            // This handler uses MouseDown instead of Click event because
            // Click won't be allowed if other field's validation fails

            this.DialogResult = DialogResult.Cancel; // Causes form to close and return Cancel result
        }

        // Precondition:  User clicked on okBtn
        // Postcondition: If invalid field on dialog, keep form open and give first invalid
        //                field the focus. Else return OK and close form.
        private void okBtn_Click(object sender, EventArgs e)
        {
            // This method does additional validation beyond what is necessary
            // in an attempt to demonstrate stronger enforcement of common
            // business rules. Name, and ID must have data entered.
            // To make this work, I chose to NOT have a DialogResult
            // property value for the okBtn.

            if (patronNameTxt.TextLength == 0) // Should not be possible but just to be safe
                patronNameTxt.Focus();
            else if (patronIdTxt.TextLength == 0) // ID box never entered, so no validation
                patronIdTxt.Focus();
            else
                this.DialogResult = DialogResult.OK; // Causes form to close and return OK result
        }
    }
}
```

Book Form

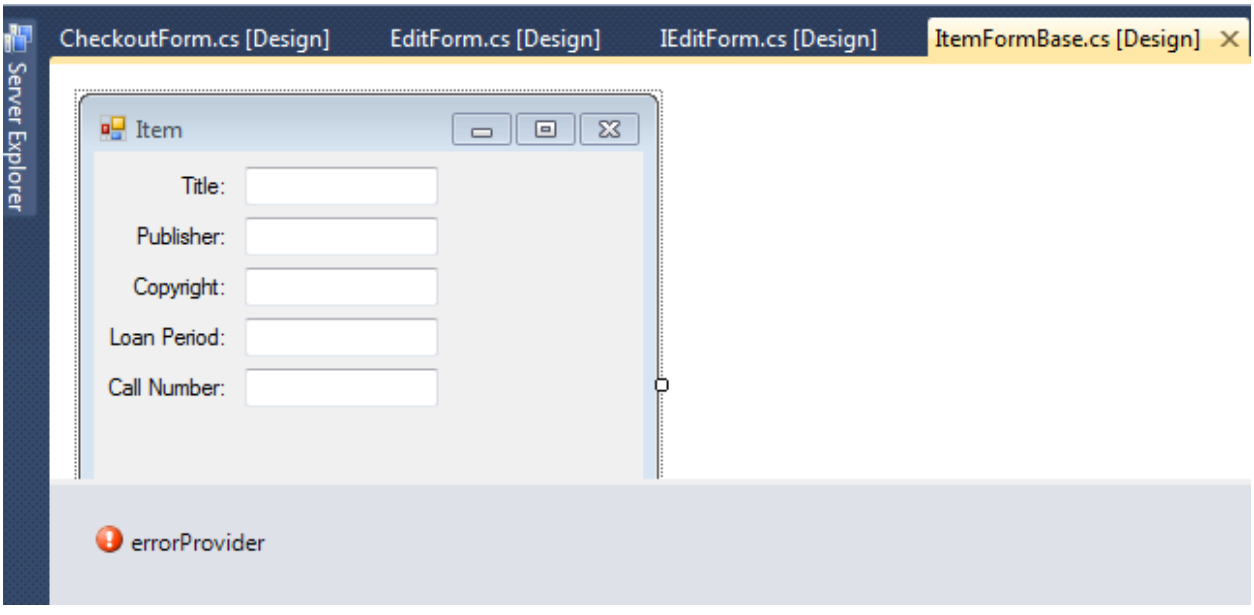


Book Form Code

```
// File: BookForm.cs
// This class creates the Book dialog box form GUI. It performs validation
// and provides String properties for each field.
// BookForm IS-A ItemFormBase

using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;

namespace LibraryItems
{
    public partial class BookForm : ItemFormBase
    {
        // Precondition:  None
        // Postcondition: The form's GUI is prepared for display.
        public BookForm()
        {
            InitializeComponent();
        }

        public String BookAuthor
        {
            // Precondition:  None
            // Postcondition: The text of form's author field has been returned
            get
            {
                return bookAuthorTxt.Text;
            }

            // Precondition:  None
```

```
            // Postcondition: The text of form's author field has been set to the specified value
            set
            {
                bookAuthorTxt.Text = value;
            }
        }

        // Precondition:  Focus is shifting from bookAuthorTxt
        // Postcondition: If text is invalid, focus remains and error provider
        //                highlights the field


        private void bookAuthorTxt_Validating(object sender, CancelEventArgs e)
        {
            if (bookAuthorTxt.TextLength == 0) // Empty field
            {
                e.Cancel = true;
                errorProvider.SetError(bookAuthorTxt, "Must provide Author");
            }
        }

        // Precondition:  Validating of bookAuthorTxt not cancelled, so data OK
        // Postcondition: Error provider cleared and focus allowed to change
        private void bookAuthorTxt_Validated(object sender, EventArgs e)
        {
            errorProvider.SetError(bookAuthorTxt, "");
        }

        // Precondition:  User clicked on okBtn
        // Postcondition: If invalid field on dialog, keep form open and give first invalid
        //                field the focus. Else return OK and close form.
        private void okBtn_Click(object sender, EventArgs e)
        {
            // This method does additional validation beyond what is necessary
            // in an attempt to demonstrate stronger enforcement of common
            // business rules. Title, Publisher, Copyright, Loan Period, Call Number,
            // and Author must all have data entered. To make this work, I chose to NOT have a
            // DialogResult property value for the okBtn.

            if (itemTitleTxt.TextLength == 0) // Should not be possible but just to be safe
                itemTitleTxt.Focus();
            else if (itemPublisherTxt.TextLength == 0)
                itemPublisherTxt.Focus();
            else if (itemCopyrightTxt.TextLength == 0)
                itemCopyrightTxt.Focus();
            else if (itemLoanPeriodTxt.TextLength == 0)
                itemLoanPeriodTxt.Focus();
            else if (itemCallNumberTxt.TextLength == 0)
                itemCallNumberTxt.Focus();
            else if (bookAuthorTxt.TextLength == 0)
                bookAuthorTxt.Focus();
            else
                this.DialogResult = DialogResult.OK; // Causes form to close and return OK result
        }
    }
}
```

Item Form



Item Code

```
// Program 2
// CIS 200-01
// Spring 2011
// Due: 3/22/2011


// File: ItemFormBase.cs
// This class creates the base class for the Item dialog box form GUIs. It performs validation
// and provides String properties for each field.

using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;

namespace LibraryItems
{
    public partial class ItemFormBase : Form
    {
        // Precondition:  None
        // Postcondition: The form's GUI is prepared for display.
        public ItemFormBase()
        {
            InitializeComponent();
        }

        public String ItemTitle
        {
            // Precondition:  None
            // Postcondition: The text of form's title field has been returned
            get
            {
                return itemTitleTxt.Text;
            }
```

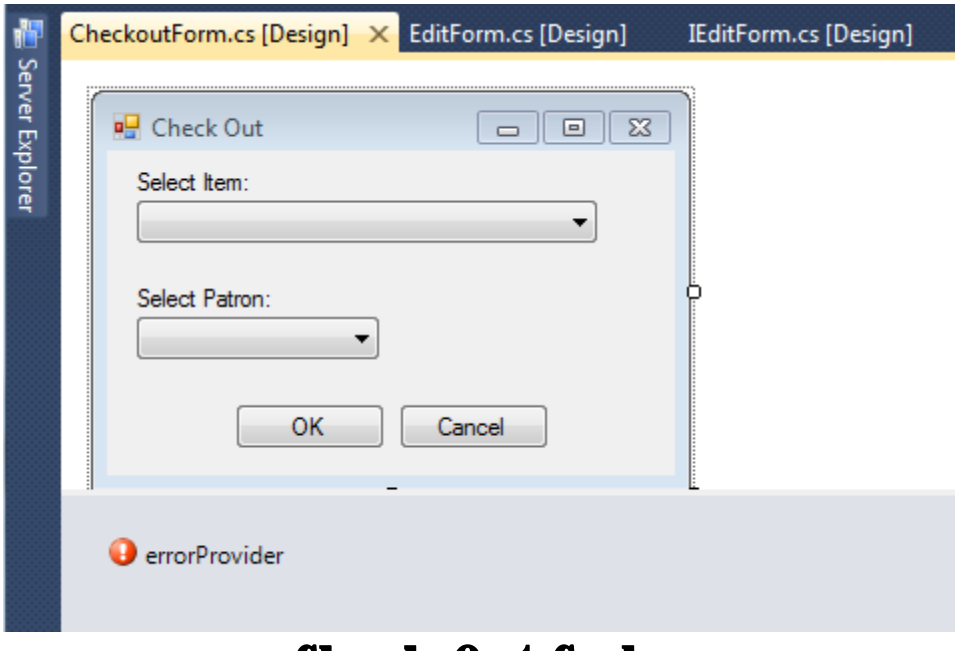
```
        // Precondition:  None
        // Postcondition: The text of form's title field has been set to the specified value
        set
        {
            itemTitleTxt.Text = value;
        }
    }

    public String ItemPublisher
    {
        // Precondition:  None
        // Postcondition: The text of form's publisher field has been returned
        get
        {
            return itemPublisherTxt.Text;
        }

        // Precondition:  None
        // Postcondition: The text of form's publisher field has been set to the specified value
        set
        {
            itemPublisherTxt.Text = value;
        }
    }

    public String ItemCopyrightYear
    {
        // Precondition:  None
        // Postcondition: The text of form's copyright field has been returned
        get
        {
            return itemCopyrightTxt.Text;
        }

        // Precondition:  None
        // Postcondition: The text of form's copyright field has been set to the specified value
        set
        {
            itemCopyrightTxt.Text = value;
        }
    }

    public String ItemLoanPeriod
    {
        // Precondition:  None
        // Postcondition: The text of form's loan period field has been returned
        get
        {
            return itemLoanPeriodTxt.Text;
        }

        // Precondition:  None
        // Postcondition: The text of form's loan period field has been set to the specified value
        set
        {
            itemLoanPeriodTxt.Text = value;
        }
    }

    public String ItemCallNumber
    {
        // Precondition:  None
```

```
    // Postcondition: The text of form's call number field has been returned
    get
    {
        return itemCallNumberTxt.Text;
    }

    // Precondition:  None
    // Postcondition: The text of form's call number field has been set to the specified value
    set
    {
        itemCallNumberTxt.Text = value;
    }
}

// Precondition:  Focus is shifting from itemTitleTxt
// Postcondition: If text is invalid, focus remains and error provider
//                highlights the field
private void itemTitleTxt_Validating(object sender, CancelEventArgs e)
{
    if (itemTitleTxt.TextLength == 0) // Empty field
    {
        e.Cancel = true;
        errorProvider.SetError(itemTitleTxt, "Must provide Title");
    }
}

// Precondition:  Validating of itemTitleTxt not cancelled, so data OK
// Postcondition: Error provider cleared and focus allowed to change
private void itemTitleTxt_Validated(object sender, EventArgs e)
{
    errorProvider.SetError(itemTitleTxt, "");
}

// Precondition:  Focus is shifting from itemPublisherTxt
// Postcondition: If text is invalid, focus remains and error provider
//                highlights the field
private void itemPublisherTxt_Validating(object sender, CancelEventArgs e)
{
    if (itemPublisherTxt.TextLength == 0) // Empty field
    {
        e.Cancel = true;
        errorProvider.SetError(itemPublisherTxt, "Must provide Publisher");
    }
}

// Precondition:  Validating of itemPublisherTxt not cancelled, so data OK
// Postcondition: Error provider cleared and focus allowed to change
private void itemPublisherTxt_Validated(object sender, EventArgs e)
{
    errorProvider.SetError(itemPublisherTxt, "");
}

// Precondition:  Focus is shifting from itemCopyrightTxt
// Postcondition: If text is invalid, focus remains and error provider
//                highlights the field
private void itemCopyrightTxt_Validating(object sender, CancelEventArgs e)
{
    int copyright;     // Copyright year of item
    bool valid = true; // Is text valid?

    if (!int.TryParse(itemCopyrightTxt.Text, out copyright)) // Parse failed?
        valid = false;
    else if (copyright < 0)
```

```
        valid = false;

    if (!valid) // Invalid, so cancel and highlight field
    {
        e.Cancel = true;
        itemCopyrightTxt.SelectAll();
        errorProvider.SetError(itemCopyrightTxt, "Invalid Copyright Year!");
    }
}

// Precondition:  Validating of itemCopyrightTxt not cancelled, so data OK
// Postcondition: Error provider cleared and focus allowed to change
private void itemCopyrightTxt_Validated(object sender, EventArgs e)
{
    errorProvider.SetError(itemCopyrightTxt, "");
}

// Precondition:  Focus is shifting from itemLoanPeriodTxt
// Postcondition: If text is invalid, focus remains and error provider
//                highlights the field
private void itemLoanPeriodTxt_Validating(object sender, CancelEventArgs e)
{
    int loanPeriod;     // Loan period of item
    bool valid = true; // Is text valid?

    if (!int.TryParse(itemLoanPeriodTxt.Text, out loanPeriod)) // Parse failed?
        valid = false;
    else if (loanPeriod < 0)
        valid = false;

    if (!valid) // Invalid, so cancel and highlight field
    {
        e.Cancel = true;
        itemLoanPeriodTxt.SelectAll();
        errorProvider.SetError(itemLoanPeriodTxt, "Invalid Loan Period!");
    }
}

// Precondition:  Validating of itemLoanPeriodTxt not cancelled, so data OK
// Postcondition: Error provider cleared and focus allowed to change
private void itemLoanPeriodTxt_Validated(object sender, EventArgs e)
{
    errorProvider.SetError(itemLoanPeriodTxt, "");
}

// Precondition:  Focus is shifting from itemCallNumberTxt
// Postcondition: If text is invalid, focus remains and error provider
//                highlights the field
private void itemCallNumberTxt_Validating(object sender, CancelEventArgs e)
{
    if (itemCallNumberTxt.TextLength == 0) // Empty field
    {
        e.Cancel = true;
        errorProvider.SetError(itemCallNumberTxt, "Must provide Call Number");
    }
}

// Precondition:  Validating of itemCallNumberTxt not cancelled, so data OK
// Postcondition: Error provider cleared and focus allowed to change
private void itemCallNumberTxt_Validated(object sender, EventArgs e)
{
    errorProvider.SetError(itemCallNumberTxt, "");
}
```

```
        // Precondition:  User pressed on cancelBtn
        // Postcondition: Form closes and sends Cancel result
        private void cancelBtn_MouseDown(object sender, MouseEventArgs e)
        {
            // This handler uses MouseDown instead of Click event because
            // Click won't be allowed if other field's validation fails

            this.DialogResult = DialogResult.Cancel; // Causes form to close and return Cancel result
        }

    }
}
```

Check Out Form



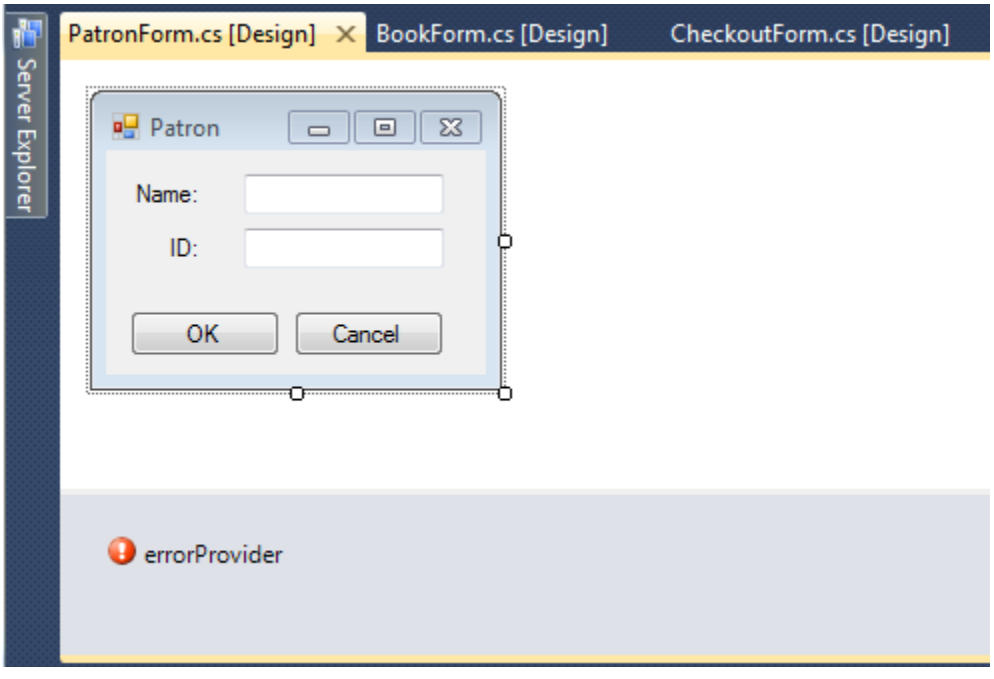
Check Out Code

```
// Program 3
// CIS 200-01
// Spring 2011
// Due: 4/6/2011
// Charles Rady

// File: CheckoutForm.cs
// This class creates the Check Out dialog box form GUI. It performs validation
// and provides int get properties for each field that are associated with the
// index of the selected item and patron.

using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;

namespace LibraryItems
{
    public partial class CheckoutForm : Form
```

```
{
    private List<LibraryItem> items;     // List of library items
    private List<LibraryPatron> patrons; // List of patrons

    // Precondition:  Lists itemList and patronList are populated with the available
    //                LibraryItems and LibraryPatrons, respectively, to choose from
    // Postcondition: The form's GUI is prepared for display.
    public CheckoutForm(List<LibraryItem> itemList, List<LibraryPatron> patronList)
    {
        InitializeComponent();

        items = itemList;
        patrons = patronList;
    }

    // Precondition:  None
    // Postcondition: The lists of items and patrons are used to populate the
    //                item and patron combo boxes, respectively
    private void CheckoutForm_Load(object sender, EventArgs e)
    {
        foreach (LibraryItem item in items)
            itemCbo.Items.Add(item.Title + ", " + item.CallNumber);

        foreach (LibraryPatron patron in patrons)
            patronCbo.Items.Add(patron.PatronName + ", " + patron.PatronID);
    }

    public int ItemIndex
    {
        // Precondition:  None
        // Postcondition: The index of form's selected item combo box has been returned
        get
        {
            return itemCbo.SelectedIndex;
        }
    }

    public int PatronIndex
    {
        // Precondition:  None
        // Postcondition: The index of form's selected patron combo box has been returned
        get
        {
            return patronCbo.SelectedIndex;
        }
    }

    // Precondition:  Focus is shifting from itemCbo
    // Postcondition: If selection is invalid, focus remains and error provider
    //                highlights the field
    private void itemCbo_Validating(object sender, CancelEventArgs e)
    {
        if (itemCbo.SelectedIndex == -1) // Nothing selected
        {
            e.Cancel = true;
            errorProvider.SetError(itemCbo, "Must select Item");
        }
    }

    // Precondition:  Validating of itemCbo not cancelled, so data OK
    // Postcondition: Error provider cleared and focus allowed to change
    private void itemCbo_Validated(object sender, EventArgs e)
    {
```

```
            errorProvider.SetError(itemCbo, "");
        }

        // Precondition:  Focus is shifting from patronCbo
        // Postcondition: If selection is invalid, focus remains and error provider
        //                highlights the field
        private void patronCbo_Validating(object sender, CancelEventArgs e)
        {
            if (patronCbo.SelectedIndex == -1) // Nothing selected
            {
                e.Cancel = true;
                errorProvider.SetError(patronCbo, "Must select Patron");
            }
        }

        // Precondition:  Validating of patronCbo not cancelled, so data OK
        // Postcondition: Error provider cleared and focus allowed to change
        private void patronCbo_Validated(object sender, EventArgs e)
        {
            errorProvider.SetError(patronCbo, "");
        }

        // Precondition:  User pressed on cancelBtn
        // Postcondition: Form closes and sends Cancel result
        private void cancelBtn_MouseDown(object sender, MouseEventArgs e)
        {
            // This handler uses MouseDown instead of Click event because
            // Click won't be allowed if other field's validation fails

            this.DialogResult = DialogResult.Cancel; // Causes form to close and return Cancel result
        }

        // Precondition:  User clicked on okBtn
        // Postcondition: If invalid field on dialog, keep form open and give first invalid
        //                field the focus. Else return OK and close form.
        private void okBtn_Click(object sender, EventArgs e)
        {
            // This method does additional validation beyond what is necessary
            // in an attempt to demonstrate stronger enforcement of common
            // business rules. Name, and ID must have data entered.
            // To make this work, I chose to NOT have a DialogResult
            // property value for the okBtn.

            if (itemCbo.SelectedIndex == -1) // Should not be possible but just to be safe
                itemCbo.Focus();
            else if (patronCbo.SelectedIndex == -1) // Patron combo never entered, so no validation
                patronCbo.Focus();
            else
                this.DialogResult = DialogResult.OK; // Causes form to close and return OK result
        }
    }
}
```

Patron Edit Form

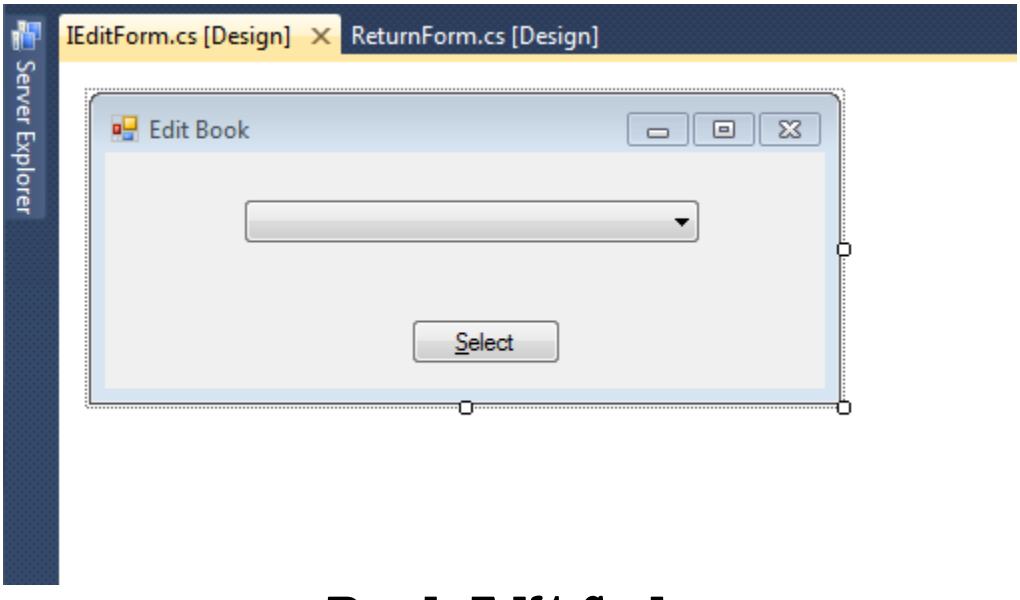


Patron Edit Code

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using System.IO;
using System.Runtime.Serialization.Formatters.Binary;
using System.Runtime.Serialization;

namespace LibraryItems
{
    public partial class EditForm : Form
    {
        private List<LibraryPatron> patrons; // List of patrons
        private int changePatron;//variable to hold the index value (an int)
        // Precondition:  None
        // Postcondition: the constructor has initialized the form and the list
        public EditForm()
        {
            InitializeComponent();// init the form
            patrons = new List<LibraryPatron>();//create the new list

        }
        // Precondition:  None
        // Postcondition: The constructor has set the values for the list and the index value. it also
inits the form
        public EditForm(List<LibraryPatron> patronList)
        {
            InitializeComponent();//init the form
            patrons = patronList;//set the list
            PatronIndex = changePatron;//set the index value
        }
```

```
        // Precondition:  The select button has been clicked
        // Postcondition: the changePatron value has been set to the index of the patron chosen by the
user
        private void selectButton_Click(object sender, EventArgs e)
        {
            int changePatron = editBox.SelectedIndex;//set the value

        }

        // Precondition:  None
        // Postcondition: The index of form's selected item combo box has been returned
        public int PatronIndex
        {
             get{return editBox.SelectedIndex;}
             set{changePatron = value;}
        }

        //precondition: none
        //postcondition: the form is loaded with the data from the list
        private void EditForm_Load(object sender, EventArgs e)
        {

            foreach (LibraryPatron item in patrons)
            editBox.Items.Add(item.PatronName);
        }
        //precondition: none
        //postcondition: the editBox is making sure that something is chosen
        private void editBox_Validating(object sender, CancelEventArgs e)
        {

            if (editBox.SelectedIndex == -1) // Nothing selected
            {
                e.Cancel = true;
                MessageBox.Show("ERROR! you must choose a patron");//show error message

            }

        }




        }

    }
```

Book Edit Form

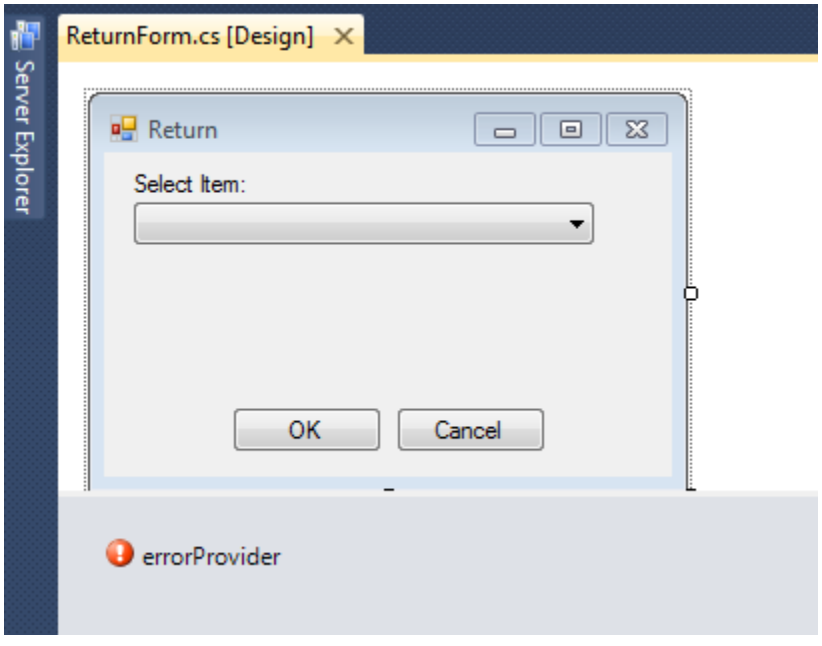


Book Edit Code

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using System.IO;
using System.Runtime.Serialization.Formatters.Binary;
using System.Runtime.Serialization;


namespace LibraryItems
{
    public partial class IEditForm : Form
    {

        private List<LibraryItem> items; // List of patrons
        private int changeItem;//variable for the index

        //precondition: none
        //postcondition: the IeditForm is initialized, the list is initialized
        public IEditForm()//constructor
        {
            InitializeComponent();//init the form
            items = new List<LibraryItem>();//list of libraryItems

        }
        //precondition: none
        //postcondition: the IeditForm is initialized, the list set and the itemIndex has been set to
equal changeItem
        public IEditForm(List<LibraryItem> itemList)
        {
            InitializeComponent();//nit the form
            items = itemList;//set items equal to itemList
            itemIndex = changeItem;//set itemIndex equal to changeItem
        }
```

```
    // Precondition:  None
    // Postcondition: The index of form's selected item combo box has been returned
    public int itemIndex
    {
        get { return editBox2.SelectedIndex; }//get accessor
        set { changeItem = value; }//set accessor
    }
    // Precondition:  the button has been clicked
    // Postcondition: changeItem has been set to the correct index
    private void button1_Click(object sender, EventArgs e)
    {

        int changeItem = editBox2.SelectedIndex;//set changeItem equal to the index

    }
    // Precondition:  None
    // Postcondition: the form has been loaded with the data from the list
    private void IEditForm_Load(object sender, EventArgs e)
    {
        foreach (LibraryBook item in items)

            editBox2.Items.Add(item.Title);//add the items
    }



    }

}
```

Return Form



Return Code

```
// Program 3
// CIS 200-01
// Spring 2011
// Due: 4/6/2011
// Charles Rady

// File: ReturnForm.cs
// This class creates the Return dialog box form GUI. It performs validation
// and provides an int get property associated with the index of the
// selected item.

using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;

namespace LibraryItems
{
    public partial class ReturnForm : Form
    {
        private List<LibraryItem> items; // List of library items

        // Precondition:  List itemList is populated with the available LibraryItems
        //                to choose from
        // Postcondition: The form's GUI is prepared for display.
        public ReturnForm(List<LibraryItem> itemList)
        {
            InitializeComponent();

            items = itemList;
        }

        // Precondition:  None
```

```
// Postcondition: The list of items is used to populate the
//                item combo box
private void ReturnForm_Load(object sender, EventArgs e)
{
    foreach (LibraryItem item in items)
        itemCbo.Items.Add(item.Title + ", " + item.CallNumber);
}

public int ItemIndex
{
    // Precondition:  None
    // Postcondition: The index of form's selected item combo box has been returned
    get
    {
        return itemCbo.SelectedIndex;
    }
}

// Precondition:  Focus is shifting from itemCbo
// Postcondition: If selection is invalid, focus remains and error provider
//                highlights the field
private void itemCbo_Validating(object sender, CancelEventArgs e)
{
    if (itemCbo.SelectedIndex == -1) // Nothing selected
    {
        e.Cancel = true;
        errorProvider.SetError(itemCbo, "Must select Item");
    }
}

// Precondition:  Validating of itemCbo not cancelled, so data OK
// Postcondition: Error provider cleared and focus allowed to change
private void itemCbo_Validated(object sender, EventArgs e)
{
    errorProvider.SetError(itemCbo, "");
}

// Precondition:  User pressed on cancelBtn
// Postcondition: Form closes and sends Cancel result
private void cancelBtn_MouseDown(object sender, MouseEventArgs e)
{
    // This handler uses MouseDown instead of Click event because
    // Click won't be allowed if other field's validation fails

    this.DialogResult = DialogResult.Cancel; // Causes form to close and return Cancel result
}

// Precondition:  User clicked on okBtn
// Postcondition: If invalid field on dialog, keep form open and give first invalid
//                field the focus. Else return OK and close form.
private void okBtn_Click(object sender, EventArgs e)
{
    // This method does additional validation beyond what is necessary
    // in an attempt to demonstrate stronger enforcement of common
    // business rules. Name, and ID must have data entered.
    // To make this work, I chose to NOT have a DialogResult
    // property value for the okBtn.

    if (itemCbo.SelectedIndex == -1) // Should not be possible but just to be safe
        itemCbo.Focus();
    else
        this.DialogResult = DialogResult.OK; // Causes form to close and return OK result
}
```

```
    }
}
```

Main Program

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Windows.Forms;

namespace LibraryItems
{
    static class Program
    {
        /// <summary>
        /// The main entry point for the application.
        /// </summary>
        [STAThread]
        static void Main()
        {
            Application.EnableVisualStyles();
            Application.SetCompatibleTextRenderingDefault(false);
            Application.Run(new Prog2Form());
        }
    }
}
```

GradeBook code

Main Program

```
// Program 4
// CIS 199-01/-76
// Due: Tuesday April 20 by class
// By: Charles Rady

// File: Program.cs
// This file serves as a simple test program for the gradebook class.
// It creates a gradebook object, uses counter-controlled repetition to allow user to
// set the number of students in this gradebook and allows instructor to enter scores
// (or not) for each student. It then produces a summary including
// number of students, number of valid scores, mean, min, max, and instructor, course,
// and assignment name
//*******LINES 91-95 TEST GetScore METHOD REMOVE "//" TO USE THIS FUNCTIONALITY*********
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace Prog3
{
    class Program
    {
        // Precondition:  None
        // Postcondition: The Gradebook class has been tested using counter controlled
        // repetition with an array to allow user to enter scores for as many students
        // as they like
        static void Main(string[] args)
        {
            Gradebook gb1;     // Test Gradebook object
            string course;     // Course name
            string instructor; // Instructor name
            string assignment; // Assignment name
            double score;      // Input score
            int scorecount = 0;    // for prompting by index number
            int students;       //number of students
            int single;        //for testing of GetScores method (remove "//" lines 86-90)



            Console.WriteLine("Welcome to the Gradebook Application\n");  //print welcome

            Console.Write("Enter course name: "); //get user input
            course = Console.ReadLine();
            Console.Write("Enter instructor name: ");
            instructor = Console.ReadLine();
            Console.Write("Enter assignment name: ");
            assignment = Console.ReadLine();
            Console.Write("Enter the number of students: ");
            students = int.Parse(Console.In.ReadLine());

            while (students < 0) //validates number of students
            {
                Console.Write("INVALID ENTRY PLEASE TRY AGAIN");//this error if a negative value is
entered
                Console.Write("\nEnter the number of students: ");//asks for new value
```

```
                students = int.Parse(Console.In.ReadLine());//user inputs new value
            }


            gb1 = new Gradebook(course, instructor, students); // Construct the Gradebook

            Console.WriteLine("\n                    ***Enter scores for " + assignment +
"***");//instruct user
            Console.WriteLine("you have entered " + students + " students, therefore your index numbers
will be 0 - " + (students - 1));//show actual index numbers

            // Priming read
            Console.Write("\nEnter score for student " + scorecount +": ");//prompt user with index
number
            score = double.Parse(Console.ReadLine());//accept user input


            do   // will allow scores to be entered until the number of students chosen by the user has
been reached
            {

                gb1.AddScore(scorecount, score);//call method AddScore and insert values
                scorecount++;//counter
                if (scorecount < students)//checks to see if there is room in array for more values
                {
                    Console.Write("Enter score for student " + scorecount + ": ");//prompt for score
                    score = double.Parse(Console.ReadLine());
                }
                else
                    scorecount = students; //sets scorecount to students


            }
            while (scorecount < students);//keep doing while this is true



            Console.WriteLine("\nGradebook results");//show results
            Console.WriteLine("Assignment: {0}", assignment);
            Console.WriteLine(gb1.ToString());
//            Console.WriteLine("would you like to test GetScores method?");
//            Console.WriteLine("enter a student index number: 0-" + (students - 1));
//            single = int.Parse(Console.ReadLine());
//            gb1.SetScores(single);
//            Console.WriteLine("student " + single + " scored: {0:f1}", gb1.GetScores());
            Console.ReadLine();



        }
    }
}
```

GradeBook Class

```
// Program 4
// CIS 199-01/-76
// Due: Tuesday April 20 by class
// By: Charles Rady

// File: Gradebook.cs
// This file creates a simple Gradebook class capable of tracking
// the name of the course and instructor plus the sum, count, min,
// max, and mean of the scores added to it. It stores each score individually in an array
// and can reference these scores individually

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

public class Gradebook
{
    private String courseName;     // Name of the course
    private String instructorName; // Name of the instructor
    private int numStudents;         // number of students
    private int validScores;        //number of valid scores
    private double findScore;    //for use in GetScores method to find specific score in array
    private double sum;            // Running total of the scores
    private int count;             // Running count of the scores
    private double min;            // Current minimum score
    private double max;            // Current maximum score
    private double mean;           // mean of all valid scores

    public double[] studentarray; //init an array
    public const double NO_SCORE = -1.0; // Value returned when no valid score entered

    // Precondition:  None
    // Postcondition: The gradebook has been initialized with the specified
    //                values for course name and instructor name. The
    //                gradebook will be empty.
    public Gradebook(String cName, String iName, int nStudents)//constructor
    {
        sum = 0.0; // Gradebook starts out empty
        count = 0;
        validScores = 0;
        mean = 0;
        CourseName = cName; // Use property to initialize in case of validation
        InstructorName = iName;
        if (nStudents < 0)//validates that the number of students is not negative
            numStudents = 10;//sets number of students to 10 if nStudents is invalid
        else
        numStudents = nStudents;//sets number of students
        studentarray = new double[numStudents] ;//init array with numStudents
        for (int placeholder = 0; placeholder < studentarray.GetLength(0); placeholder++) //goes
through new array step by step
            studentarray[placeholder] = NO_SCORE; //sets initial values in array to NO_SCORE

        min = 101; //starting value so that first score entered will be set to min
        max = NO_SCORE;//starting value so that first valid score entered will be set to max
    }

    public String InstructorName
    {
        // Precondition:  None
        // Postcondition: The name of the instructor has been returned
```

```
    get
    {
        return instructorName;
    }

    // Precondition:  None
    // Postcondition: The name of the instructor has been set to the
    //                specified value
    set
    {
        instructorName = value;
    }
}

public String CourseName
{
    // Precondition:  None
    // Postcondition: The name of the course has been returned
    get
    {
        return courseName;
    }

    // Precondition:  None
    // Postcondition: The name of the course has been set to the
    //                specified value
    set
    {
        courseName = value;
    }
}

// Precondition:  score >= 0 and score <=100
// Postcondition: The score has been added to the array if found to be valid
public void AddScore(int studentIndex, double score)
{
    if (score >= 0 && score <= 100) // Valid score?
    {
        studentarray[studentIndex] = score; //if valid then store value
        validScores++; //add to the running count of valid scores
    }

    else // NO_SCORE to be entered

        studentarray[studentIndex] = NO_SCORE;

        ++count; //add to the running count of all entries

}


// Precondition:  None
// Postcondition: The current total of the scores added to the
//                gradebook has been returned
public double GetSum()
{
    for (int n = 0; n < numStudents; n++)//runs through array
        if(studentarray[n] > 0) //prevent NO_SCORE from being calculated
        sum = sum + studentarray[n];//adds to the sum
    return sum;
}

// Precondition:  None
```

```
// Postcondition: the mean of valid scores has been returned

public double GetMean()
{
    for (int n = 0; n < numStudents; n++)//runs through array
    {
        if (studentarray[n] > -1)//prevent NO_SCORE from being calculated
         sum = sum + studentarray[n]; //adds to the sum

        mean = (sum / validScores);//finds mean

    }
    return mean;
}

// Precondition:  none
// Postcondition: The current minimum score has been returned
//
public double GetMin()
{
    for (int n = 0; n < numStudents; n++)//runs through array to find min

        if (studentarray[n] < min && studentarray[n] > NO_SCORE)      // New smallest?
            min = studentarray[n];      // if so then set min to new value

    return min;
}

//precondition: none
//postcondition: the current maximum score has been returned
public double GetMax()
{
    for (int n = 0; n < numStudents; n++)//runs through array to find max

        if (studentarray[n] > max)      // New largest?
            max = studentarray[n];      // if so then set max to new value


    return max;
}
//precondition: none
//postcondition: the number of students has been returned
public int GetNumStudents()
{
    return numStudents;
}
// Precondition: none
// Postcondition: returns (integer) number of valid scores entered
public int GetValidScores()
{
    return validScores;
}

// Precondition: none
// Postcondition: sets where to look in the array
public void SetScores(int index)
{
    if (index > 0 && index < studentarray.GetLength(0))//check validity
        findScore = studentarray[index];//set findScore value
    else
        findScore = -1;
}
// Precondition: none
```

```
    // Postcondition: returns value from array
    public double GetScores()
    {
        return findScore;
    }


    public override string ToString()
    {
        String gradeStr; // Formatted output string

        if (GetValidScores() == 0) // No valid scores
            gradeStr = String.Format("Course: {0}\n" +
            "Instructor: {1}\nnumber of students: {2}\n" +
            "Mean: {3}\nMin: {3}\n" +
            "Max: {3}", CourseName, InstructorName, GetNumStudents(),
             "Undefined");
        else
        {
            gradeStr = String.Format("Course: {0}\n" +
               "Instructor: {1}" + "\nNumber of Students: {2}" +"\nValid Scores: {3}\n" +
               "Mean: {4:F1}\nMin: {5:F1}\n" +
               "Max: {6:F1}", CourseName, InstructorName, GetNumStudents(),
               GetValidScores(), GetMean(), GetMin(), GetMax());

        }

        return gradeStr;
    }
}
```

LINQ exercise

Note: this test program requires the Library class hierarchy to run.

```
//PROGRAM 1A BUILT FROM
// Program 1 (Instructor's solution)
// CIS 200-01
// Due: 2/21/2011
// 1A by Charles Rady
// File: Program.cs
// This file creates a simple test program that builds objects of each
// of the concrete classes from the LibraryItem hierarchy. It does so using the List collection.
// it then sends the results of LINQ queries to the console
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using LibraryItems;
public class Program
{
// Precondition: None
// Postcondition: The LibraryItem hierarchy has been tested
public static void Main(string[] args)
{
const int DAYSLATE = 14; // Number of days late to test each object's CalcLateFee method
//create a new list of LibraryItem objects and LibraryPatron objects
List<LibraryItem> libItems = new List<LibraryItem>();
List<LibraryPatron> libPatrons = new List<LibraryPatron>();
//create the actual objects
// note, these steps can be combined as libItems.Add (new LibraryBook("The Wright Guide to C#", "UofL
Press", 2010, 14,
// "ZZ25 3G", "Andrew Wright")); but then I can't figure out how to reach them once they are in the
list
LibraryBook b1 = new LibraryBook("The Wright Guide to C#", "UofL Press", 2010, 14,
"ZZ25 3G", "Andrew Wright"); // Test book
LibraryBook b2 = new LibraryBook("The Rady Guide to C# bumbling", "cdrady Press", 2010, 14,
"ZZ27 3G", "Charles Rady"); // Test book);
LibraryMovie m1 = new LibraryMovie("Andrew's Super-Duper Movie", "UofL Movies", 2011, 7,
"MM33 2D", 92.5, "Andrew L. Wright", LibraryMediaItem.MediaType.BLURAY, LibraryMovie.MPAARatings.PG);
// Test movie
LibraryMovie m2 = new LibraryMovie("Star Wars", "LucasFilm", 1977, 7,
"MM35 3D", 97.5, "George Lucas", LibraryMediaItem.MediaType.BLURAY, LibraryMovie.MPAARatings.PG); //
Test movie
LibraryMusic t1 = new LibraryMusic("C# - The Album", "UofL Music", 2011, 14,
"CD44 4Z", 84.3, "Dr. A", LibraryMediaItem.MediaType.CD, 10); // Test music item
```

```
LibraryMusic t2 = new LibraryMusic("Alive", "cdradyproductions.com", 208, 14,
"CD46 4Z", 84.3, "CDRadyrealmusic", LibraryMediaItem.MediaType.CD, 10); // Test music item
LibraryJournal j1 = new LibraryJournal("Journal of C# Goodness", "UofL Journals", 2011, 14,
"JJ12 7M", 1, 2, "Information Systems", "Andrew Wright"); // Test journal
LibraryJournal j2 = new LibraryJournal("Journal of Db Goodness", "UofL Journals", 2011, 14,
"JJ15 7M", 1, 2, "Information Systems", "Andrew Wright"); // Test journal
LibraryMagazine z1 = new LibraryMagazine("C# Monthly", "UofL Mags", 2010, 14,
"MA53 9A", 16, 9); // Test magazine
LibraryMagazine z2 = new LibraryMagazine("C# Monthly", "UofL Mags", 2010, 14,
"MA57 9A", 19, 12); // Test magazine
//add LibraryItems objects to List of LibraryItems
libItems.Add (b1); // Test book
libItems.Add(b2); // Test book
libItems.Add(m1); // test movie
libItems.Add(m2); // test movie
libItems.Add(t1); // test tunes
libItems.Add(t2); // test tunes
libItems.Add(j1); //test journal
libItems.Add(j2); //test journal
libItems.Add(z1); //test magazine
libItems.Add(z2); // test magazine
//create patrons
// note, these steps can be combined as libPatrons.Add(new LibraryPatron("Ima Bad Reader", "12365"));
but then I can't figure out
// how to reach them once they arein the list
LibraryPatron p1 = new LibraryPatron("Ima Reader", "12345"); // Test patron
LibraryPatron p2 = new LibraryPatron("Ima Bad Reader", "12365"); // Test patron
LibraryPatron p3 = new LibraryPatron("Ima Good Reader", "12325"); // Test patron
LibraryPatron p4 = new LibraryPatron("Ima PC", "12315"); // Test patron
LibraryPatron p5 = new LibraryPatron("Ima MAC", "12395"); // Test patron
//add 5 patrons to the patron list
libPatrons.Add(p1);
libPatrons.Add(p2);
libPatrons.Add(p3);
libPatrons.Add(p4);
libPatrons.Add(p5);
//display each item in the LibraryItem list
foreach(var libItem in libItems)
Console.Write("\n {0}\n", libItem);
//pause until user hits a key
Console.Write("\n\n\thit enter to continue");
Console.ReadLine();
//checkout 5 items
b1.CheckOut(p1);
m1.CheckOut(p2);
t2.CheckOut(p3);
j2.CheckOut(p4);
z1.CheckOut(p5);
```

```
//display each item in the LibraryItem list again
foreach (var libItem in libItems)
Console.Write("\n {0}\n", libItem);
//pause until user hits a key
Console.Write("\n\n\thit enter to continue");
Console.ReadLine();
//use linq to select checked out items
var checkedout =
from l in libItems
where l.itemCheckedOut == true
select l;
//display the checked out items
foreach (var m in checkedout)
Console.Write(m);
Console.Write("\n\n\tCount of checked out items:\t {0}", checkedout.Count());
//pause until user hits a key
Console.Write("\n\n\thit enter to continue");
Console.ReadLine();
//use linq to select only the media items that are checked out
var mediaout =
from m in checkedout
where m.CallNumber.StartsWith("C") || m.CallNumber.StartsWith("MM")
select m;
//display those media items that are checked out
foreach (var m in mediaout)
Console.Write("{0}\n\n", m);
//pause until user hits a key
Console.Write("\n\n\thit enter to continue");
Console.ReadLine();
//use LINQ to select the titles of LibraryMagazine objects
var magazine =
from n in libItems
where n.CallNumber.StartsWith("MA")
select n.Title;
//display the unique titles of magazines in the list
foreach (var element in magazine.Distinct())
Console.Write("{0}",element);
//pause until user hits a key
Console.Write("\n\n\thit enter to continue");
Console.ReadLine();
//calculate late fee for each item in the list and display it with title and call number
foreach(var element in libItems)
Console.WriteLine( "{0}\t{1}\nAfter {2} days, owe {3:C}\n",element.Title, element.CallNumber, DAYSLATE,
element.CalcLateFee(DAYSLATE));
//pause until user hits a key
Console.Write("\n\n\thit enter to continue");
Console.ReadLine();
```

```
//return the items
b1.ReturnToShelf();
m1.ReturnToShelf();
t2.ReturnToShelf();
j2.ReturnToShelf();
z1.ReturnToShelf();
//display the count of checked out items with the linq result variable
Console.Write("\n\n\n\tCount of checked out items:\t {0}", checkedout.Count());
//pause until user hits a key
Console.Write("\n\n\thit enter to continue");
Console.ReadLine();
//select library books that are in the list and their loan periods
var libraryBook =
from l in libItems
where l.CallNumber.StartsWith("ZZ")
select new { l.Title, LoanPeriod = l.LoanPeriod };
//display these books and their loan periods
foreach (var element in libraryBook)
Console.Write("\n\n{0}",element);
//modify the LibraryBooks in the list to add 7 days to their loan periods
b1.LoanPeriod = b1.LoanPeriod + 7;
b2.LoanPeriod = b2.LoanPeriod + 7;
//display the books with the new loan periods
Console.Write("\n\n\tRevised Loan Periods");
foreach (var element in libraryBook)
Console.Write("\n\n{0}", element);
//pause until user hits a key
Console.Write("\n\n\thit enter to continue");
Console.ReadLine();
//display entire list again
foreach (var libItem in libItems)
Console.Write("\n {0}\n", libItem);
//pause until user hits a key
Console.Write("\n\n\thit enter to continue");
Console.ReadLine();
}
}
```

Sort Example

(Library hierarchy required)

MAIN PROGRAM

```
// Program 4
// CIS 200-01
// Due: 4/18/2011
// By: Charles Rady
// File: Program.cs
// This file creates a simple test program that builds objects of each
// of the concrete classes from the LibraryItem hierarchy. It also creates a list of the objects and
sorts them in various ways
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using LibraryItems;
public class Program
{
// Precondition: None
// Postcondition: The LibraryItem hierarchy has been tested
public static void Main(string[] args)
{
List<LibraryItem> items = new List<LibraryItem>(); //list to hold the library items
// Test data - Magic numbers allowed here
items.Add(new LibraryBook("The Wright Guide to C#", "UofL Press", 2010, 14,
"ZZ25 3G", "Andrew Wright")); // Test book
items.Add(new LibraryMovie("Andrew's Super-Duper Movie", "UofL Movies", 2011, 7,
"MM33 2D", 92.5, "Andrew L. Wright", LibraryMediaItem.MediaType.BLURAY,
LibraryMovie.MPAARatings.PG)); // Test movie
items.Add(new LibraryMusic("C# - The Album", "UofL Music", 2011, 14,
"CD44 4Z", 84.3, "Dr. A", LibraryMediaItem.MediaType.CD, 10)); // Test music item
items.Add(new LibraryJournal("Journal of C# Goodness", "UofL Journals", 2011, 14,
"JJ12 7M", 1, 2, "Information Systems", "Andrew Wright")); // Test journal
items.Add(new LibraryMagazine("C# Monthly", "UofL Mags", 2017, 14,
"MA53 9A", 16, 9)); // Test magazine
items.Add(new LibraryBook("Rady Guide to C# Bumbling", "UofL Press", 2000, 14,
"ZZ25 3G", "Andrew Wright")); // Test book
items.Add(new LibraryMovie("Chuck's not-so-much Movie", "UofL Movies", 2001, 7,
"MM33 2D", 92.5, "Andrew L. Wright", LibraryMediaItem.MediaType.BLURAY,
LibraryMovie.MPAARatings.PG)); // Test movie
items.Add(new LibraryMusic("excerpts from: C# - The Album", "UofL Music", 2001, 14,
"CD44 4Z", 84.3, "Dr. A", LibraryMediaItem.MediaType.CD, 10)); // Test music item
items.Add(new LibraryJournal("Journal of Java Goodness", "UofL Journals", 2011, 14,
```

```
"JJ12 7M", 1, 2, "DisInformation Systems", "Andrew Wright")); // Test journal
items.Add(new LibraryMagazine("Java Monthly", "UofL Mags", 2010, 14,
"MA53 9A", 16, 9)); // Test magazine
//set the window size and position
Console.SetWindowPosition(0,0);//set the console to the top left corner
Console.SetWindowSize(55, 60);//set the console size to fit the objects it is designed for
//Display the list unsorted
Console.WriteLine("**Displaying title only**\n\nNormal Order\nUnsorted\n\n");//write to console
foreach (LibraryItem item in items)//step through list
Console.WriteLine(item);//write the object titles to the console
Console.WriteLine("\n\nHit enter to continue");//display message
Console.ReadLine();//pause for user input
//sorted (natural order)
items.Sort(); // Sort - uses natural order
Console.Out.WriteLine("Sorted list (natural order)\nSorted by title\n\n");//display message
foreach (LibraryItem item in items)//step through list
Console.WriteLine(item);//display each item
Console.WriteLine("\n\nHit enter to continue");//display message
Console.ReadLine();//pause for user input
//sorted (reverse order by copyright year)
items.Sort(new reversesorter()); // Sort - uses specified Comparer class
Console.WriteLine("Sorted list (reverse natural order) using Comparer:\nSorted by copyright year,
descending\n\n");//display message
foreach (LibraryItem Item in items)//step through list
Console.WriteLine(Item);//display results
Console.WriteLine("\n\nhit enter to see extra credit attempt");//display message
Console.ReadLine();//pause for user input
//EXTRA CREDIT
// will re-submitif I can get this to work right
items.Sort();
foreach (LibraryItem item in items)
{
Console.WriteLine (item.GetType());
}
Console.WriteLine("hit enter to exit");//display message
Console.ReadLine();//wait for user input
}//end main method
}//end program
```

REVERSE SORT CLASS

```
//CHARLES RADY - PROGRAM 4
//This class sorts in reverse order by copyright year
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
namespace LibraryItems
{
public class reversesorter : IComparer<LibraryItem>
{
// Precondition: None
// Postcondition: When item1 < item2, method returns negative #
// When item1 > item2, method returns positive #
// When neither are true == must be true - sets to 0
public int Compare(LibraryItem itemA, LibraryItem itemB)
{
LibraryItem item1 =(LibraryItem)itemA;//local variable
LibraryItem item2 = (LibraryItem)itemB;//local variable
if (item1.CopyrightYear < item2.CopyrightYear)//check for less than
return 1;
if (item1.CopyrightYear > item2.CopyrightYear)//check for greater than
return -1;
else
return 0;//return 0 if neither (==)
}//end method
}//end class
}//end namespace
```

THIS PROGRAM REQUIRES THE LIBRARY HEIRARCHY TO RUN
THE LIBRARYITEM CLASS MUST INCLUDE THE ICOMPARABLE INTERFACE

SQL: Queries

```
--Charles Rady--
--Assignment 8--
--due 11/1/2010--

--1--
select distinct
P.paintID,P.colorList,orderDate,B.modelType,B.saleState,C.customerID,C.lastName,C.firstNa
me
from customer C INNER JOIN bicycle B ON C.customerID = B.customerID
     inner join Paint P on P.PaintID=B.PaintID
                  Where modelType like ('%Mountain%')
                  AND
                  saleState='CA'
                  AND colorList like ('%RED%')
                  AND
                  B.orderDate >= '2003-09-01' and orderdate <='2003-09-30'


--2--
select B.storeID,B.saleState,E.employeeID, B. orderDate,E.lastName,B.modelType
from bicycle B inner join employee E on B.employeeID = E.employeeID
      where storeID=2
      AND
      B.saleState='WI'
      AND
      modelType='Race'
      AND
      B.orderDate >= '2001-01-01' and B.orderDate <='2001-12-31'

--3--
select distinct C.componentID,M.manufacturerName,productNumber
from component C inner join manufacturer M on C.manufacturerID=M.manufacturerID
                         inner join bikeParts BP on BP.componentID=C.componentID
                         inner join bicycle B on B.serialNumber=BP.serialNumber
where C.category like ('Rear Derailleur%') and Road='Road' and B.saleState='FL'
                         and year(orderDate)=2002

--4--
select B.frameSize, C.firstName, C.lastName, B.saleState, B.orderDate, B.modelType
from bicycle B inner join customer C on B.customerID = C.customerID
where
            B.saleState='GA'
            and
            B.orderDate>= '2004-01-01' and B.orderDate <='2004-12-31'
            and
            modelType='Mountain Full'
            and
            B.frameSize=21


--5--
Select      m.manufacturerID, m.manufacturerName
From  manufacturer m inner join purchaseOrder p on m.manufacturerID = p.manufacturerID
Where p.discount =
                              (
                              Select max (discount)
                              From purchaseOrder
                              Where year(orderDate) = 2003
                                    )
```

```
--6--
select C. listPrice,C. quantityOnHand,C. road,M. manufacturerName
from component C inner join manufacturer M on C.manufacturerID = M.manufacturerID
where

ListPrice=(select max(ListPrice)
                  from component
                  where
                  quantityOnHand>200
                  and
                  Road='Road'
             )
      Group By quantityOnHand,listPrice,road,manufacturerName
      having (quantityOnHand)>200

--7--
Select C.componentID, M.manufacturerName, C.productNumber, C.category, C.year,
(quantityOnHand * estimatedCost) as Value
From component C inner join manufacturer M on M.manufacturerID = C.manufacturerID
Where (quantityOnHand * estimatedCost) =
         (
         Select max (quantityOnHand *estimatedCost)
         From component
           )

--8--
Select BP.employeeID, lastName, count(BP.componentID)
From bikeParts BP inner join employee E on BP.employeeID = E.employeeID
Group by BP.employeeID, lastName, dateInstalled
Having count(BP.componentID) =
            (
            Select top 1 count(BP.ComponentID)
            From bikeParts BP
            Group by dateInstalled
            Order by count(BP.componentID) desc
            )


--9--
Select letterStyleID, count (letterStyleID) as CountOfSerialNumber
From  bicycle
where modelType = 'Race' and  year(orderDate) = 2003
group by letterStyleID
having count(letterStyleID) =
            (
            Select top 1 count(letterStyleID)
            From bicycle
            Where modelType = 'Race' and year(orderDate) = 2003
            Group by letterStyleID
            Order by count(letterStyleID) desc
            )



--10--
Select b.customerID,lastName,firstName, count(serialNumber) as NumOfBikes,
sum(b.saleprice) as AmtSpent
From bicycle b inner join customer c on b.customerID = c.customerID
where year(orderDate) = 2002
group by b.customerID, lastName, firstName
```

```
having sum(saleprice) =
      (
      Select top 1 sum(salePrice)
      From bicycle b inner join customer c on b.customerID = c.customerID
    where year(orderDate) = 2002
    group by b.customerID, lastName, firstName
    Order By sum(salePrice) desc
      )



--11--
Select year(orderDate) as SaleYear, count(SerialNumber) as CountOfSerialNumber
From Bicycle
Where ModelType like '%mountain%'

and year(orderDate) between 2000 and 2004
Group by year(orderDate)
Order by year(orderDate)asc


--12--

Select c.componentID, m.manufacturerID, sum(p.pricePaid * p.quantity) as Value
From component c inner join manufacturer m on m.manufacturerID = c.manufacturerID
                                  inner join purchaseItem p on p.componentID =
c.componentID
                                  inner join purchaseOrder po on po.purchaseID =
p.purchaseID
                                 where year(po.orderDate) = 2003
group by c.componentID, m.manufacturerID
having sum(p.pricePaid * p.quantity) = (select TOP 1 SUM(p.pricePaid * p.quantity)

From component c inner join manufacturer m on m.manufacturerID = c.manufacturerID
                                  inner join purchaseItem p on p.componentID =
c.componentID
                                  inner join purchaseOrder po on po.purchaseID =
p.purchaseID
                                 where year(po.orderDate) = 2003
group by c.componentID, m.manufacturerID
Order By Sum(p.pricePaid * p.quantity) DESC)


--13--
Select e.employeeID, lastName, count(b.painter) as NumberPainted
From employee e inner join bicycle b on e.employeeID = b.painter
                          inner join paint p on p.paintID = b.paintID
Where modelType = 'Race' and year(orderDate) = 2003
      and month(orderDate) = 5 and colorList like '%red%'
      Group by e.employeeID, lastName
      Having count(b.painter) =
            (
            Select top 1 count(painter) as code
            From bicycle b inner join paint p on p.paintID = b.paintID
            Where modelType = 'race' and year(orderDate) = 2003 and month(orderDate) = 5
and colorList like '%red%'
            Group by painter
            Order by code desc
            )


--14--
```

```
Select rs.storeID, storeName, city, sum(salePrice) as SumOfSalePrice
From retailStore rs inner join city c on rs.cityID = c.cityID
inner join bicycle b on b.storeID = rs.storeID
Where state = 'CA' and year(orderDate) = 2003
Group by rs.storeID, storeName, city
having sum(salePrice) =
      (
      Select top 1 sum(salePrice) as code
      From retailStore rs inner join city c on rs.cityID = c.cityID
      inner join bicycle b on b.storeID = rs.storeID
      Where state = 'CA' and year(orderDate) = 2003
      Group by rs.storeID
      Order by code desc
      )


--15--
Select sum(quantity * weight) as TotalComponentWeight
From bicycle b inner join bikeparts bp on b.serialNumber = bp.serialNumber
       inner join component c on c.componentID = bp.componentID

Where bp.serialNumber = 11356




--16--
Select sum(listPrice) as SumOfListPrices_CampyRecords
From groupo g inner join groupComponents gc on g.componentgroupID = gc.groupID
       inner join component c on gc.componentID = c.componentID

Where groupName = 'Campy Record 2002'


--17--
Select material, count(b.serialNumber) as CountOfSerialNumber
From bicycle b inner join bicycleTubeUsage bu on b.serialNumber = bu.serialNumber
       inner join tubeMaterial tm on bu.tubeID = tm.tubeID
       inner join bikeTubes bt on bt.serialNumber = b.serialNumber

Where material = 'carbon fiber' and year(startDate) = 2003 and tubeName = 'Down' and
modelType = 'Race'
OR material = 'titanium' and year(startDate) = 2003 and tubeName = 'Down' and modelType =
'Race'
Group by material


--18--
select AVG(PricePaid) as Avg_of_Price_Paid
from PurchaseItem P inner join Component C on P.ComponentID = C.ComponentID
where P.ComponentID=313600

--19--
select AVG(TopTube) as AVG_TopTube_Length_1999
from Bicycle B inner join BikeTubes BT on B.SerialNumber=BT.SerialNumber
where B.FrameSize=54 and YEAR(B.StartDate)=1999

--20--
select AVG(ListPrice)as Mountain_row1_Road_row2
from Component,ComponentName
where ComponentName='Rear tire' and Road='MTB'
union
```

```
select AVG(ListPrice)
from Component,ComponentName
where ComponentName='Rear tire' and Road='Road'

--21--
Select *
From bicycle b inner join employee e on b.employeeID = e.employeeID
Where year(startDate) = 2003 and month(StartDate)=5 and modelType = 'road'


--22--
Select p.PaintID, p.ColorName, Count(b.serialNumber) as NumberOfBikesPainted
From paint p inner join bicycle b on p.paintID = b.paintID
       inner join letterStyle ls on ls.letterStyle = b.letterStyleID
Where year(orderDate) = 2002 and letterStyleID like '%English%'
Group by ColorName, p.paintID
Order by NumberofBikesPainted desc

--23--
Select serialNumber, modelType, orderDate, salePrice
From bicycle b
Where year(orderDate) = 2003 and modelType = 'Race' and saleprice >
        (
         Select avg(salePrice) as AvgPriceOfRaceBikes
         From bicycle
         Where year(orderDate) = 2002 and modelType = 'Race'
         )
Order by orderDate desc


--24--

Select max(c.componentID) as highestinventoryvalue
From bicycle b inner join bikeParts bp on b.serialNumber = bp.serialNumber
                     inner join component c on c.componentID = bp.componentID
Where year(orderDate) = 2004
--and DateInstalled=null
--I tried creating a view to work from that (view is CDRADY01 and is still stored in my
database)
-- but didn't get anywhere with this one


--25--
Select (manufacturerName) as Store_or_Manufacturer_Name, m.Phone
From manufacturer m inner join city c on m.cityID = c.cityID
Union
Select storeName, rs.phone
From city c inner join retailStore rs on c.cityID = rs.cityID
              inner join bicycle b on b.StoreID = rs.StoreID
Where year(orderDate) = 2004 and state = 'ca'


--26--
Select (m.lastName)as ManagerLastName, e.employeeID, e.lastName, e.firstName, e.Title
From employee e inner join employee m on e.currentManager = m.employeeID
Where m.lastName = 'venetiaan'


--27--
Create view componentsused as
Select componentID, SUM(quantity) as totalqtyused
From bike..bikeParts
```

```
where dateInstalled < '6/30/2000'
group by componentID

Create View componentspurchased AS
Select PI.componentID, SUM(PI.quantity) AS totalqtypurchased
From bike..purchaseItem PI INNER JOIN bike..purchaseOrder PO ON PI.purchaseID =
PO.purchaseID
Where PO.orderDate < '6/30/2000'
group by PI.componentID

--views created--
--execute the following in CIS31034 database--
Select c.componentID, totalqtyused,totalqtypurchased
from componentspurchased c,componentsused cc
where totalqtypurchased >=(totalqtyused *1.25)
order by totalqtyused
--date constraint is already in the views
```

SQL: Star Schema/Data Warehouse

```
--Charles Rady
--Assignment 11
--This is the create tables code followed by the stored procedure, followed by the 3 queries
--due Wed dec 8
--PART 1: CREATE THE TABLES--

 --CREATE TABLES--
Create table RepDIM
(
Rep_Key INT identity (1,1),
Rep_Num int,
Last_Name nvarchar(15),
First_Name nvarchar(15),
Street nvarchar(15),
City nvarchar(15),
State nvarchar(2),
Zip varchar(5),
Commission money,
Rate float
)

 Create Table CustomerDIM
(
Customer_Key INT identity (1,1),
Customer_Num nvarchar(3) not null,
Customer_Name nvarchar(35) not null,
Street nvarchar(15) not null,
City nvarchar(15) not null,
State nvarchar(2) not null,
Zip nvarchar(5) not null,
Balance money not null,
Credit_Limit money not null,
Rep_Num nvarchar(2)not null
)


Create Table PartDIM
(
Part_Key INT identity(1,1),
Part_Num nvarchar(4) not null,
Description nvarchar(15) not null,
On_Hand int not null,
Class nvarchar(2)not null,
Warehouse nvarchar(1) not null,
```

```
Price money not null
)
Create Table TimeDIM
(
Time_Key INT identity(1,10),
Ord_Date datetime NOT NULL
)
Create Table FACT
(
Part_Key int NOT NULL,
Customer_Key int NOT NULL,
Rep_Key int NOT NULL,
Time_Key int NOT NULL,
list_Price money NOT NULL,
Quantity int NOT NULL,
ORDER_DATE DATETIME
)
--ALTER TABLE KEYS, add PKs. FKs are removed and re-added by the stored procedure, tables are truncated
--whenever it is run
Alter table RepDIM
add constraint pk_RepDIM primary key (Rep_Key)
Alter table CustomerDIM
add constraint pk_CustomerDIM primary key (Customer_Key)
Alter table PartDIM
add constraint pk_PartDIM primary key(Part_Key)
Alter table TimeDIM
add constraint pk_TimeDIM primary key (Time_Key)
Alter table FACT
add constraint fk_Fact_RepDIM foreign key (Rep_Key) references RepDIM (Rep_Key)
Alter table FACT
add constraint fk_Fact_CustomerDIM foreign key (Customer_Key) references CustomerDIM (Customer_Key)
Alter table FACT
add constraint fk_Fact_PartDIM foreign key (Part_Key) references PartDIM (Part_Key)
Alter table FACT
add constraint fk_Fact_TimeDIM foreign key (Time_Key) references TimeDIM (Time_Key)

--create staging table
CREATE TABLE STAGING
(
Rep_Key INT,
Customer_Key int,
Part_Key int,
Time_Key int,
Customer_Num int,
Part_Num varchar(4),
Rep_Num int,
On_Hand float,
Order_Date datetime,
Price money
)
```

```
--PART 2 STORED PROCEDURE--
USE [cis31034]
GO
/****** Object: StoredProcedure [dbo].[A11] Script Date: 12/08/2010 09:25:49 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
-- =============================================
-- Author: <Author,,Name>
-- Create date: <Create Date,,>
-- Description: <Description,,>
-- =============================================
ALTER PROCEDURE [dbo].[A11]
-- Add the parameters for the stored procedure here (NEXT PAGE)
```

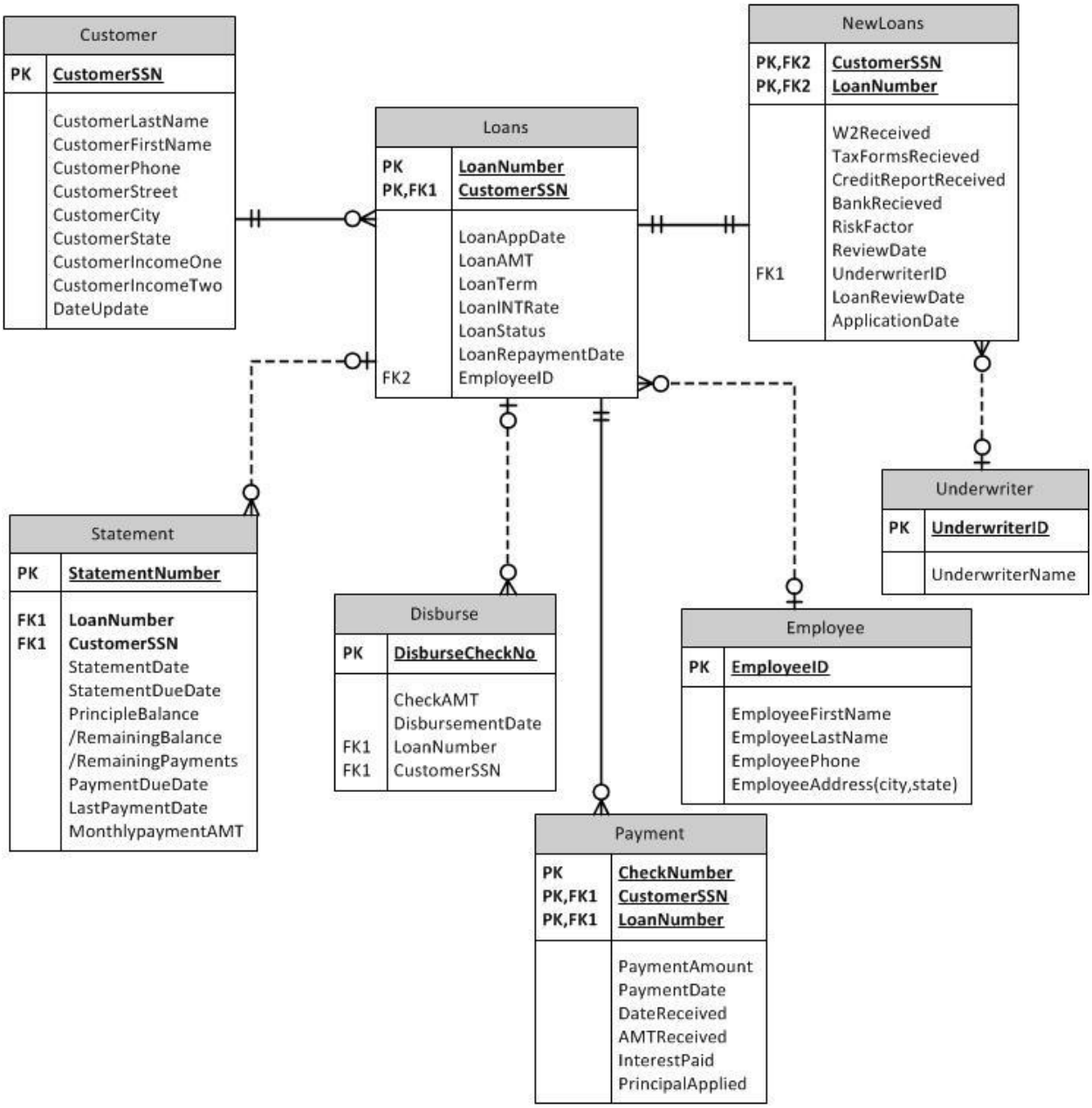
```
AS
BEGIN
-- SET NOCOUNT ON added to prevent extra result sets from
-- interfering with SELECT statements.
SET NOCOUNT ON;

--drop any previous FK constraints from FACT table if they exist
IF EXISTS (SELECT * FROM sys.objects WHERE object_id = OBJECT_ID(N'[dbo].[fk_Fact_RepDIM]'))
BEGIN
ALTER TABLE FACT
DROP CONSTRAINT fk_Fact_RepDIM
END
IF EXISTS (SELECT * FROM sys.objects WHERE object_id = OBJECT_ID(N'[dbo].[fk_Fact_PartDIM]'))
BEGIN
ALTER TABLE FACT
DROP CONSTRAINT fk_Fact_PartDIM
END
IF EXISTS (SELECT * FROM sys.objects WHERE object_id = OBJECT_ID(N'[dbo].[fk_Fact_TimeDIM]'))
BEGIN
ALTER TABLE FACT
DROP CONSTRAINT fk_Fact_TimeDIM
END
IF EXISTS (SELECT * FROM sys.objects WHERE object_id = OBJECT_ID(N'[dbo].[fk_Fact_CustomerDIM]'))
BEGIN
ALTER TABLE FACT
DROP CONSTRAINT fk_Fact_CustomerDIM
END

--clean out existing data in database warehouse tables
BEGIN
truncate table RepDIM;
truncate table TimeDIM;
truncate table PartDIM;
truncate table FACT;
truncate table STAGING;
truncate table CustomerDIM;
END

--POPULATE DIMENSION TABLES
BEGIN
insert into RepDIM
Select *
From Rep
insert into CustomerDIM
Select *
From Customer
insert into partDIM
Select *
```

```
from part
insert into TimeDIM
select Order_Date
from ORDERS
END

--add foreign key constraints
BEGIN
Alter table FACT
add constraint fk_Fact_RepDIM foreign key (Rep_Key) references RepDIM (Rep_Key)
Alter table FACT
add constraint fk_Fact_CustomerDIM foreign key (Customer_Key) references CustomerDIM (Customer_Key)
Alter table FACT
add constraint fk_Fact_PartDIM foreign key (Part_Key) references PartDIM (Part_Key)
Alter table FACT
add constraint fk_Fact_TimeDIM foreign key (Time_Key) references TimeDIM (Time_Key)
END
--populate staging table

INSERT INTO STAGING(Customer_Num, Part_Num, Rep_Num, On_Hand, order_date, Price)
SELECT C.CUSTOMER_NUM, P.PART_NUM,R.REP_NUM,P.ON_HAND AS quantity,o.Order_Date, P.PRICE AS
List_Price
FROM CUSTOMER C INNER JOIN REP R ON C.REP_NUM=R.REP_NUM
INNER JOIN ORDERS O ON O.CUSTOMER_NUM=C.CUSTOMER_NUM
INNER JOIN ORDER_LINE OL ON OL.ORDER_NUM=O.ORDER_NUM
INNER JOIN PART P ON P.PART_NUM=OL.PART_NUM

BEGIN
--populate DIM keys w/update statement
update staging
set rep_key = r.rep_key
from repdim r inner join staging s on r.rep_NUM = S.rep_NUM
update staging
set time_key = t.time_key
from timedim t inner join staging s on t.ORD_DATE = S.ORDER_DATE
update staging
set Part_key = p.part_key
from Partdim p inner join staging s on p.part_NUM = S.part_NUM
update staging
```

```
set Customer_key = C.Customer_key
from Customerdim c inner join staging s on c.customer_NUM = S.customer_NUM
end

--populate FACT table
insert into FACT(Part_Key,Customer_Key,Rep_Key,Time_Key,List_Price,quantity,ORDER_DATE)
SELECT PART_KEY,CUSTOMER_KEY,REP_KEY,TIME_KEY,PRICE,ON_HAND,ORDER_DATE
from STAGING
END

--part 3, queries

--# 1
--all sales are in october in this database
SELECT (quantity * list_price) as avg_sale_amt,order_date,c.customer_name
FROM FACT F INNER JOIN CUSTOMERDIM C ON F.CUSTOMER_KEY=C.CUSTOMER_KEY
ORDER BY ORDER_DATE desc
--group by order_date
--having order_date >= 8/01/2003

--#2
select p.Part_key,Description,month(f.order_date)as Month
from FACT F INNER JOIN PartDIM P on f.part_key=p.part_key
inner join TimeDIM T on F.Time_Key = T.Time_Key
group by p.Part_key,Description,month(order_date)
having count(p.part_key)>1

--#3
SELECT quantity as totalSold,rep_num,city
FROM FACT F INNER JOIN CUSTOMERDIM C ON F.CUSTOMER_KEY=C.CUSTOMER_KEY
ORDER BY ORDER_DATE desc
```

SQL: ERD/Business Rules

Assignment 9 ERD



**when used in a database, LoanNumber should be an auto-generated number

Assignment A9 Business Rules

1. A customer may have zero to many loans but each loan may be assigned to exactly one customer.
2. New loan applications are associated with one, and only one loan, and a loan is associated with zero to many new loan applications.
3. A loan can have one and only one underwriter, while an underwriter can be associated with zero or many new loans.
4. An employee will work on zero or many loans, but a loan can be associated with one employee.
5. A payment applies to one and only one loan.
6. A loan can have zero, or many payments.
7. Disbursement checks apply to either one loan, or none, while a loan can have zero, or many disbursements.
8. A loan statement is associated exactly one loan
9. Loans can have zero, or many statements.

PHP code from cdradyproductions.com

The “Examples page” at cdradyproductions.com/cis/localarticle4

```
<?php include("password_protect.php"); ?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">

<head>
<meta content="en-us" http-equiv="Content-Language" />
<meta content="text/html; charset=utf-8" http-equiv="Content-Type" />
<title>EXAMPLES</title>
<style type="text/css">
.auto-style5 {
                                                color: #000000;
}
.auto-style8 {
                                                font-size: small;
}
.auto-style10 {
                                                text-align: left;
                                                font-family: "Gill Sans", "Gill Sans MT", Calibri, "Trebuchet MS", sans-serif;
                                                font-size: large;
}
.auto-style12 {
                                                margin-top: 0px;
                                                margin-right: 0px;
}
.auto-style19 {
                                                font-size: large;
                                                font-family: "Gill Sans", "Gill Sans MT", Calibri, "Trebuchet MS", sans-serif;
}
.auto-style39 {
                                                font-size: medium;
}
.auto-style7 {
                                                text-decoration: none;
}
.auto-style42 {
                                                font-size: x-large;
                                                font-family: "Gill Sans", "Gill Sans MT", Calibri, "Trebuchet MS", sans-serif;
}
.auto-style43 {
                                                font-size: x-large;
}
.auto-style45 {
                                                text-align: left;
                                                border-bottom-style: solid;
                                                border-bottom-width: 1px;
}
.auto-style46 {
                                                border-bottom-style: solid;
                                                border-bottom-width: 1px;
}
.auto-style47 {
                                                border-top-style: solid;
                                                border-top-width: 1px;
                                                border-bottom-style: solid;
                                                border-bottom-width: 1px;
```

```
}
.auto-style48 {
                                font-family: "Gill Sans", "Gill Sans MT", Calibri, "Trebuchet MS", sans-serif;
                                border-top-style: solid;
                                border-top-width: 1px;
                                border-bottom-style: solid;
                                border-bottom-width: 1px;
}
.auto-style51 {
                                font-size: x-large;
                                font-family: "Gill Sans", "Gill Sans MT", Calibri, "Trebuchet MS", sans-serif;
                                text-decoration: underline;
}
.auto-style52 {
                                text-decoration: underline;
}
.auto-style53 {
                                text-align: left;
                                font-family: "Gill Sans", "Gill Sans MT", Calibri, "Trebuchet MS", sans-serif;
                                font-size: large;
                                text-decoration: underline;
                                border-bottom-style: solid;
                                border-bottom-width: 1px;
}
.auto-style18 {
                                font-family: "Gill Sans", "Gill Sans MT", Calibri, "Trebuchet MS", sans-serif;
}
.auto-style40 {
                                color: #000000;
                                font-size: large;
}
.auto-style4 {
                                text-align: left;
}
.auto-style21 {
                                font-family: Impact, Haettenschweiler, "Arial Narrow Bold", sans-serif;
                                color: rgb(255, 255, 255);
}
.auto-style57 {
                                text-decoration: underline;
                                font-size: x-large;
}
.auto-style58 {
                                text-align: center;
}
.auto-style59 {
                                margin-left: 0px;
}
.auto-style60 {
                                color: #000000;
                                font-size: medium;
}
.auto-style61 {
                                text-decoration: underline;
                                font-family: "Gill Sans", "Gill Sans MT", Calibri, "Trebuchet MS", sans-serif;
}
.auto-style62 {
                                text-align: left;
                                font-size: medium;
}
.auto-style63 {
                                font-size: large;
}
.auto-style65 {
                                color: #000000;
                                font-family: "Times New Roman", Times, serif;
```

```
}
.auto-style66 {
                                                text-decoration: underline;
                                                font-family: "Gill Sans", "Gill Sans MT", Calibri, "Trebuchet MS", sans-serif;
                                                font-size: large;
}
.auto-style67 {
                                                font-size: x-small;
                                                border-bottom-style: none;
}
.auto-style68 {
                                                font-size: x-small;
}
.auto-style69 {
                                                text-decoration: underline;
                                                font-size: large;
}
</style>
<script type="text/javascript">
<!--
function FP_swapImg() {//v1.0
 var doc=document,args=arguments,elm,n; doc.$imgSwaps=new Array(); for(n=2; n<args.length;
 n+=2) { elm=FP_getObjectByID(args[n]); if(elm) { doc.$imgSwaps[doc.$imgSwaps.length]=elm;
 elm.$src=elm.src; elm.src=args[n+1]; } }
}

function FP_preloadImgs() {//v1.0
 var d=document,a=arguments; if(!d.FP_imgs) d.FP_imgs=new Array();
 for(var i=0; i<a.length; i++) { d.FP_imgs[i]=new Image; d.FP_imgs[i].src=a[i]; }
}

function FP_getObjectByID(id,o) {//v1.0
 var c,el,els,f,m,n; if(!o)o=document; if(o.getElementById) el=o.getElementById(id);
 else if(o.layers) c=o.layers; else if(o.all) el=o.all[id]; if(el) return el;
 if(o.id==id || o.name==id) return o; if(o.childNodes) c=o.childNodes; if(c)
 for(n=0; n<c.length; n++) { el=FP_getObjectByID(id,c[n]); if(el) return el; }
 f=o.forms; if(f) for(n=0; n<f.length; n++) { els=f[n].elements;
 for(m=0; m<els.length; m++){ el=FP_getObjectByID(id,els[n]); if(el) return el; } }
 return null;
}
// -->
</script>
</head>

<body style="background-color: #B2B2B2"
onload="FP_preloadImgs(/*url*/'button14.gif',/*url*/'button15.gif',/*url*/'button51.gif',/*url*/'button52.gif',/*url*/'button55.gif',/*url*/'button57
.gif',/*url*/'button58.gif',/*url*/'button60.gif',/*url*/'button61.gif',/*url*/'button62.gif',/*url*/'button63.gif',/*url*/'button64.gif',/*url*/'button67.gif'
,/*url*/'button68.gif',/*url*/'button5C.gif',/*url*/'button5D.gif',/*url*/'button70.gif',/*url*/'button71.gif',/*url*/'button72.gif',/*url*/'button73.gif',/*
url*/'button74.gif',/*url*/'button75.gif',/*url*/'button6E.gif',/*url*/'button6F.gif',/*url*/'button76.gif',/*url*/'button77.gif',/*url*/'button78.gif',/*url*
/'button79.gif',/*url*/'button81.gif',/*url*/'button82.gif',/*url*/'button7A.gif',/*url*/'button7B.gif',/*url*/'button7D.gif',/*url*/'button7E1.gif',/*url*/'
button83.gif',/*url*/'button84.gif',/*url*/'button87.gif',/*url*/'button88.gif',/*url*/'button89.gif',/*url*/'button8A.gif',/*url*/'button8C.gif',/*url*/'but
ton8D.gif',/*url*/'button7F.gif',/*url*/'button80.gif',/*url*/'buttonA0.gif',/*url*/'buttonA1.gif',/*url*/'buttonB0.gif',/*url*/'buttonB1.gif',/*url*/'butto
n3F.gif',/*url*/'button40.gif',/*url*/'button42.gif',/*url*/'button43.gif',/*url*/'button48.gif',/*url*/'button49.gif',/*url*/'button9D.gif',/*url*/'button9
E.gif',/*url*/'buttonA3.gif',/*url*/'buttonA4.gif',/*url*/'buttonA9.gif',/*url*/'buttonAA.gif',/*url*/'buttonAC.gif',/*url*/'buttonAD.gif',/*url*/'buttonD
B.gif',/*url*/'buttonDC.gif',/*url*/'buttonE7.jpg',/*url*/'buttonE8.jpg',/*url*/'buttonEB.gif',/*url*/'buttonEC.gif',/*url*/'button11.gif',/*url*/'button
12.gif',/*url*/'button103.gif',/*url*/'button104.gif',/*url*/'button106.gif',/*url*/'button107.gif',/*url*/'button109.gif',/*url*/'buttonC1.gif',/*url*/'but
ton111.gif',/*url*/'button1A1.gif',/*url*/'button113.gif',/*url*/'button114.gif',/*url*/'button115.gif',/*url*/'button116.gif',/*url*/'button119.gif',/*ur
l*/'buttonC2.gif')">

<p class="auto-style58" style="height: 212px">
&amp;nbsp;<img alt="" height="236" src="BANNER3.gif" width="1118" class="auto-style59" /></p>
<table style="width: 120%; height: 35px;">
                                                <tr>
                                                <td class="auto-style53" style="width: 352px"><strong>System Analysis
                                                &amp;amp; Design, Development&amp;nbsp; </strong></td>
                                                 <td class="auto-style53" style="width: 472px">
```

```
                                        <strong>Object-Oriented Programming</strong></td>
                                        </tr>
</table>
<table style="width: 105%" class="auto-style12">
                                        <tr>
                                        <td class="auto-style45" style="width: 299px; height: 440px;"><strong>
                                        <span class="auto-style39">Elaboration Phase</span><br class="auto-style39" />
                                        </strong><span class="auto-style8"><em>full report including UML
diagrams,</em></span><em><br class="auto-style8" />
                                        </em><span class="auto-style8"><em>narratives,
                                        charts, feasibility report, etc. </em></span><em>
                                                <br class="auto-style8" />
                                        </em>for: Hand in Hand Ministries<br>**large file. It contains the
                                        entire project.<span class="auto-style39"><strong><br />
                                            <a href="files/Elaboration%20Phase.pdf">
                                          <img id="img50" alt="Download" height="27"
onmousedown="FP_swapImg(1,0,/*id*/'img50',/*url*/'buttonC2.gif')" onmouseout="FP_swapImg(0,0,/*id*/'img50',/*url*/'button120.gif')"
onmouseover="FP_swapImg(1,0,/*id*/'img50',/*url*/'button119.gif')" onmouseup="FP_swapImg(0,0,/*id*/'img50',/*url*/'button119.gif')"
src="button120.gif" style="border: 0" width="135" fp-style="fp-btn: Embossed Capsule 7; fp-font-size: 14; fp-bgcolor: #C0C0C0; fp-
transparent: 1" fp-title="Download" /></a><br> <br>SalesForce manual PDF </strong></span><strong><span class="auto-
style8"><em><br />
                                                    </em> </span>
                                                    </strong><em>
                                        <span class="auto-style8">The following beta-test user manual </span><br class="auto-
style8" />
                                        <span class="auto-style8">has been adopted for use as a template by
                                                    <br />
                                        U of L College of Business, CIS 420</span></em><strong><br />
                                        </strong>for: West Louisville Community Ministries<strong><br />
                                                <a href="files/SCRIPT.pdf">
                                          <img id="img1" alt="Download" height="27"
onmousedown="FP_swapImg(1,0,/*id*/'img1',/*url*/'button15.gif')" onmouseout="FP_swapImg(0,0,/*id*/'img1',/*url*/'button13.gif')"
onmouseover="FP_swapImg(1,0,/*id*/'img1',/*url*/'button14.gif')" onmouseup="FP_swapImg(0,0,/*id*/'img1',/*url*/'button14.gif')"
src="button13.gif" style="border: 0" width="135" fp-style="fp-btn: Embossed Capsule 7; fp-font-size: 14; fp-bgcolor: #C0C0C0; fp-
transparent: 1" fp-title="Download" /></a>
                                            <br> <br class="auto-style57" />
                                        <span class="auto-style51">VISIO files</span><br />
                                        <span class="auto-style39">ERD for Hand in Hand ministries, iteration 5<br />
                                                <a href="files/ERD320_2.vsd">
                                          <img id="img24" alt="Download" height="27"
onmousedown="FP_swapImg(1,0,/*id*/'img24',/*url*/'button68.gif')" onmouseout="FP_swapImg(0,0,/*id*/'img24',/*url*/'button90.gif')"
onmouseover="FP_swapImg(1,0,/*id*/'img24',/*url*/'button67.gif')" onmouseup="FP_swapImg(0,0,/*id*/'img24',/*url*/'button67.gif')"
src="button90.gif" style="border: 0" width="135" /><!-- MSComment="ibutton" fp-style="fp-btn: Embossed Capsule 7; fp-font-size: 14;
fp-bgcolor: #C0C0C0; fp-transparent: 1" fp-title="Download" --></a>
                                        </span> <a class="auto-style7" href="files/ERD320_2.jpg"><img id="img31"
alt="JPEG" height="23" onmousedown="FP_swapImg(1,0,/*id*/'img31',/*url*/'buttonA1.gif')"
onmouseout="FP_swapImg(0,0,/*id*/'img31',/*url*/'button9F.gif')" onmouseover="FP_swapImg(1,0,/*id*/'img31',/*url*/'buttonA0.gif')"
onmouseup="FP_swapImg(0,0,/*id*/'img31',/*url*/'buttonA0.gif')" src="button9F.gif" style="border: 0" width="75" /><!--
MSComment="ibutton" fp-style="fp-btn: Embossed Capsule 7; fp-font: Agency FB; fp-font-style: Bold; fp-transparent: 1; fp-proportional:
0" fp-title="JPEG" --></a><br />
                                        <span class="auto-style8"> </span><br />
                                        <span class="auto-style39">Class Diagram for Hand in Hand (it5)<br />
                                          <a href="files/ClassDiagramIteration5.vsd">
                                          <img id="img25" alt="Download" height="27"
onmousedown="FP_swapImg(1,0,/*id*/'img25',/*url*/'button5D.gif')" onmouseout="FP_swapImg(0,0,/*id*/'img25',/*url*/'button91.gif')"
onmouseover="FP_swapImg(1,0,/*id*/'img25',/*url*/'button5C.gif')" onmouseup="FP_swapImg(0,0,/*id*/'img25',/*url*/'button5C.gif')"
src="button91.gif" style="border: 0" width="135" /><!-- MSComment="ibutton" fp-style="fp-btn: Embossed Capsule 7; fp-font-size: 14;
fp-bgcolor: #C0C0C0; fp-transparent: 1" fp-title="Download" --></a> 

                                        <a class="auto-style7" href="files/ClassDiagramIteration5.jpg">
                                            <img id="img32" alt="JPEG" height="23"
onmousedown="FP_swapImg(1,0,/*id*/'img32',/*url*/'buttonB1.gif')" onmouseout="FP_swapImg(0,0,/*id*/'img32',/*url*/'buttonAF.gif')"
onmouseover="FP_swapImg(1,0,/*id*/'img32',/*url*/'buttonB0.gif')" onmouseup="FP_swapImg(0,0,/*id*/'img32',/*url*/'buttonB0.gif')"
src="buttonAF.gif" style="border: 0" width="75" /><!-- MSComment="ibutton" fp-style="fp-btn: Embossed Capsule 7; fp-font: Agency FB;
fp-font-style: Bold; fp-transparent: 1; fp-proportional: 0" fp-title="JPEG" --></a></span></strong><br>
```

```
                                        <br><strong><span class="auto-style39">ERD for Assignment #9, CIS
310</span></strong><br>
                                                <strong>
                                              <span class="auto-style39">
                                        <a class="auto-style7" href="files/A9%20+%20BR.vsd">
                                          <img id="img48" alt="Download" height="27"
onmousedown="FP_swapImg(1,0,/*id*/'img48',/*url*/'button114.gif')" onmouseout="FP_swapImg(0,0,/*id*/'img48',/*url*/'button117.gif')"
onmouseover="FP_swapImg(1,0,/*id*/'img48',/*url*/'button113.gif')" onmouseup="FP_swapImg(0,0,/*id*/'img48',/*url*/'button113.gif')"
src="button117.gif" style="border: 0" width="135" fp-style="fp-btn: Embossed Capsule 7; fp-font-size: 14; fp-bgcolor: #C0C0C0; fp-
transparent: 1" fp-title="Download" /></a><a href="files/ClassDiagramIteration5.vsd" class="auto-style7">  </a>
                                        <a class="auto-style7" href="files/9ERD_BR.pdf">
                                            <img id="img49" alt="PDF" height="23"
onmousedown="FP_swapImg(1,0,/*id*/'img49',/*url*/'button116.gif')" onmouseout="FP_swapImg(0,0,/*id*/'img49',/*url*/'button118.gif')"
onmouseover="FP_swapImg(1,0,/*id*/'img49',/*url*/'button115.gif')" onmouseup="FP_swapImg(0,0,/*id*/'img49',/*url*/'button115.gif')"
src="button118.gif" style="border: 0" width="75" fp-style="fp-btn: Embossed Capsule 7; fp-font: Agency FB; fp-font-style: Bold; fp-
transparent: 1; fp-proportional: 0" fp-title="PDF" /></a></span></strong><br />
                                                <br />
                                                <strong>
                                        <span class="auto-style51">PowerPoint presentations</span><br />
                                              Powerpoint Demo for WLCM <br />
                                        <a href="files/demo%20intro%20REVISED.pptx">
                                          <img id="img10" alt="Download" height="27"
onmousedown="FP_swapImg(1,0,/*id*/'img10',/*url*/'buttonE8.jpg')" onmouseout="FP_swapImg(0,0,/*id*/'img10',/*url*/'buttonE4.jpg')"
onmouseover="FP_swapImg(1,0,/*id*/'img10',/*url*/'buttonE7.jpg')" onmouseup="FP_swapImg(0,0,/*id*/'img10',/*url*/'buttonE7.jpg')"
src="buttonE4.jpg" style="border: 0" width="135" /><!-- MSComment="ibutton" fp-style="fp-btn: Embossed Capsule 7; fp-font-size: 14;
fp-bgcolor: #C0C0C0" fp-title="Download" --></a></strong><br />
                                                </td>
                                        <td style="width: 313px; height: 440px;" class="auto-style46"><strong>
                                        <span class="auto-style39">Library version 1</span><br />
                                        </strong><span class="auto-style8">This small GUI app allows you to view
                                              </span> <br class="auto-style8" />
                                        <span class="auto-style8">a pre-set library which contains library items
                                              </span> <br class="auto-style8" />
                                        <span class="auto-style8">and patrons. You can add patrons </span>
                                                <br class="auto-style8" />
                                        <span class="auto-style8">and books as well as check them in and out.
                                                </span>
                                              <br class="auto-style8" />
                                            <strong><a href="files/Library.exe">
                                           <img id="img40" alt=".EXE" height="29"
onmousedown="FP_swapImg(1,0,/*id*/'img40',/*url*/'buttonAD.gif')" onmouseout="FP_swapImg(0,0,/*id*/'img40',/*url*/'buttonAB.gif')"
onmouseover="FP_swapImg(1,0,/*id*/'img40',/*url*/'buttonAC.gif')" onmouseup="FP_swapImg(0,0,/*id*/'img40',/*url*/'buttonAC.gif')"
src="buttonAB.gif" style="border: 0" width="75" /><!-- MSComment="ibutton" fp-style="fp-btn: Embossed Capsule 7; fp-font-size: 14; fp-
bgcolor: #C0C0C0; fp-transparent: 1; fp-proportional: 0" fp-title=".EXE" --></a><a href="files/library1code.pdf"><img id="img41"
alt="code" height="29" onmousedown="FP_swapImg(1,0,/*id*/'img41',/*url*/'buttonDC.gif')"
onmouseout="FP_swapImg(0,0,/*id*/'img41',/*url*/'buttonDA.gif')" onmouseover="FP_swapImg(1,0,/*id*/'img41',/*url*/'buttonDB.gif')"
onmouseup="FP_swapImg(0,0,/*id*/'img41',/*url*/'buttonDB.gif')" src="buttonDA.gif" style="border: 0" width="85" /><!--
MSComment="ibutton" fp-style="fp-btn: Embossed Capsule 7; fp-font-size: 14; fp-bgcolor: #C0C0C0; fp-transparent: 1; fp-proportional:
0" fp-title="code" --></a></strong><br class="auto-style8" />
                                                <br class="auto-style8" />
                                        <strong><span class="auto-style39">Library version 2(GUI app)</span></strong><br
class="auto-style8" />
                                        <span class="auto-style8">This small app allows you to save and open
                                              </span> <br class="auto-style8" />
                                        <span class="auto-style8">library files which contain library items
                                              </span> <br class="auto-style8" />
                                        <span class="auto-style8">and patrons. You can add and edit patrons
                                              </span> <br class="auto-style8" />
                                        <span class="auto-style8">and books as well as check them in and out.
                                                </span>
                                              <br class="auto-style8" />
                                        <span class="auto-style8">*note: This download comes in a zipped folder,</span><br
class="auto-style8" />
                                        <span class="auto-style8">first open the library program and then open</span><br
class="auto-style8" />
```

```
                                        <span class="auto-style8">the test file from within the library in order</span><br
class="auto-style8" />
                                        <span class="auto-style8">to see test data to edit and checkout</span><strong><br />
                                            <a href="files/Library2.exe">
                                        <img id="img39" alt=".EXE" height="29"
onmousedown="FP_swapImg(1,0,/*id*/'img39',/*url*/'buttonAA.gif')" onmouseout="FP_swapImg(0,0,/*id*/'img39',/*url*/'buttonA8.gif')"
onmouseover="FP_swapImg(1,0,/*id*/'img39',/*url*/'buttonA9.gif')" onmouseup="FP_swapImg(0,0,/*id*/'img39',/*url*/'buttonA9.gif')"
src="buttonA8.gif" style="border: 0" width="75" /><!-- MSComment="ibutton" fp-style="fp-btn: Embossed Capsule 7; fp-font-size: 14; fp-
bgcolor: #C0C0C0; fp-transparent: 1; fp-proportional: 0" fp-title=".EXE" --></a><a href="files/Library%20Code.pdf"><img id="img37"
alt="code" height="29" onmousedown="FP_swapImg(1,0,/*id*/'img37',/*url*/'buttonA4.gif')"
onmouseout="FP_swapImg(0,0,/*id*/'img37',/*url*/'buttonA2.gif')" onmouseover="FP_swapImg(1,0,/*id*/'img37',/*url*/'buttonA3.gif')"
onmouseup="FP_swapImg(0,0,/*id*/'img37',/*url*/'buttonA3.gif')" src="buttonA2.gif" style="border: 0" width="85" /><!--
MSComment="ibutton" fp-style="fp-btn: Embossed Capsule 7; fp-font-size: 14; fp-bgcolor: #C0C0C0; fp-transparent: 1; fp-proportional:
0" fp-title="code" --></a><br />
                                                <br />
                                        <span class="auto-style5">GradeBook (Console App)
                                                </span>
                                                <br />
                                        </strong><span class="auto-style8">This simple gradebook app calculates
                                                <br />
                                            grades of n number of students and
                                                <br />
                                        calculates the mean, min and max grades <br />
                                        as well as the count of valid grades</span><br />
                                            <strong><a href="files/GradeBook.exe">
                                             <img id="img3" alt=".EXE" height="29"
onmousedown="FP_swapImg(1,0,/*id*/'img3',/*url*/'button52.gif')" onmouseout="FP_swapImg(0,0,/*id*/'img3',/*url*/'button50.gif')"
onmouseover="FP_swapImg(1,0,/*id*/'img3',/*url*/'button51.gif')" onmouseup="FP_swapImg(0,0,/*id*/'img3',/*url*/'button51.gif')"
src="button50.gif" style="border: 0" width="75" /><!-- MSComment="ibutton" fp-style="fp-btn: Embossed Capsule 7; fp-font-size: 14; fp-
bgcolor: #C0C0C0; fp-transparent: 1; fp-proportional: 0" fp-title=".EXE" --></a>

                                            <a href="files/GradeBook%20code.pdf">
                                             <img id="img36" alt="code" height="29"
onmousedown="FP_swapImg(1,0,/*id*/'img36',/*url*/'button9E.gif')" onmouseout="FP_swapImg(0,0,/*id*/'img36',/*url*/'button9C.gif')"
onmouseover="FP_swapImg(1,0,/*id*/'img36',/*url*/'button9D.gif')" onmouseup="FP_swapImg(0,0,/*id*/'img36',/*url*/'button9D.gif')"
src="button9C.gif" style="border: 0" width="85" /><!-- MSComment="ibutton" fp-style="fp-btn: Embossed Capsule 7; fp-font-size: 14; fp-
bgcolor: #C0C0C0; fp-transparent: 1; fp-proportional: 0" fp-title="code" --></a></strong><br>
                                        <br><strong><span class="auto-style51">Smaller Exercises</span><span class="auto-
style69"><br>
                                        </span></strong><span class="auto-style68">(</span><span class="auto-style67">these
                                        require the Library Class Heirarchy to run)</span><br />
                                        <strong>Linq Exercise (Code)<br><a href="files/LINQ%20exercise.pdf">
                                             <img id="img35" alt="code" height="28"
onmousedown="FP_swapImg(1,0,/*id*/'img35',/*url*/'button49.gif')" onmouseout="FP_swapImg(0,0,/*id*/'img35',/*url*/'button47.gif')"
onmouseover="FP_swapImg(1,0,/*id*/'img35',/*url*/'button48.gif')" onmouseup="FP_swapImg(0,0,/*id*/'img35',/*url*/'button48.gif')"
src="button47.gif" style="border: 0" width="85" /><!-- MSComment="ibutton" fp-style="fp-btn: Embossed Capsule 7; fp-font-size: 14; fp-
bgcolor: #C0C0C0; fp-transparent: 1; fp-proportional: 0" fp-title="code" --></a></strong><br><strong>Sort Exercise (Code)<br><a
href="files/Sort%20Example.pdf">
                                             <img id="img47" alt="code" height="28"
onmousedown="FP_swapImg(1,0,/*id*/'img47',/*url*/'button1A1.gif')" onmouseout="FP_swapImg(0,0,/*id*/'img47',/*url*/'button112.gif')"
onmouseover="FP_swapImg(1,0,/*id*/'img47',/*url*/'button111.gif')" onmouseup="FP_swapImg(0,0,/*id*/'img47',/*url*/'button111.gif')"
src="button112.gif" style="border: 0" width="85" fp-style="fp-btn: Embossed Capsule 7; fp-font-size: 14; fp-bgcolor: #C0C0C0; fp-
transparent: 1; fp-proportional: 0" fp-title="code" /></a></strong><br />
                                                </td>
                                        </tr>
                                        <tr>
                                        <td style="width: 299px; height: 386px;" class="auto-style47">
                                                <strong>
                                        <span class="auto-style42"><span class="auto-style52">Papers</span> </span>
                                            <span class="auto-style19"><br />
                                        <span class="auto-style52">research papers:</span></span><br />
                                        <span class="auto-style5">Apple ICC-ID headache </span><br />
                                                <a href="files/ATT.pdf">
                                            <img id="img9" alt="Download" height="27"
onmousedown="FP_swapImg(1,0,/*id*/'img9',/*url*/'button7E1.gif')" onmouseout="FP_swapImg(0,0,/*id*/'img9',/*url*/'button7C.gif')"
onmouseover="FP_swapImg(1,0,/*id*/'img9',/*url*/'button7D.gif')" onmouseup="FP_swapImg(0,0,/*id*/'img9',/*url*/'button7D.gif')"
```

src="button7C.gif" style="border: 0" width="135" /><!-- MSComment="ibutton" fp-style="fp-btn: Embossed Capsule 7; fp-font-size: 14; fp-bgcolor: #C0C0C0; fp-transparent: 1" fp-title="Download" -->
Standards and Practices for Non-Profits

<!-- MSComment="ibutton" fp-style="fp-btn: Embossed Capsule 7; fp-font-size: 14; fp-bgcolor: #C0C0C0; fp-transparent: 1" fp-title="Download" -->

published papers
The ev<big><big><big><big><small><small><big><big><big>olution of Gaming in America

<!-- MSComment="ibutton" fp-style="fp-btn: Embossed Capsule 7; fp-font-size: 14; fp-bgcolor: #C0C0C0; fp-transparent: 1" fp-title="View" --><big>

</big><big>

How to photoshop for free on a mac

<!-- MSComment="ibutton" fp-style="fp-btn: Embossed Capsule 7; fp-font-size: 14; fp-bgcolor: #C0C0C0; fp-transparent: 1" fp-title="View" -->

</big><big>

Dueling Pianist:
what you need to get started</big></big></big></big></small></small></big></big></big></big>

<!-- MSComment="ibutton" fp-style="fp-btn: Embossed Capsule 7; fp-font-size: 14; fp-bgcolor: #C0C0C0; fp-transparent: 1" fp-title="View" --></td>
<td style="width: 313px; height: 386px;" class="auto-style48">
SQL
Assignment 7 code (CIS 310, database design)

<img id="img6" alt="Download" height="27" onmousedown="FP_swapImg(1,0,/*id*/'img6',/*url*/'button6F.gif')" onmouseout="FP_swapImg(0,0,/*id*/'img6',/*url*/'button6D.gif')" onmouseover="FP_swapImg(1,0,/*id*/'img6',/*url*/'button6E.gif')" onmouseup="FP_swapImg(0,0,/*id*/'img6',/*url*/'button6E.gif')"

src="button6D.gif" style="border: 0" width="135" /><!-- MSComment="ibutton" fp-style="fp-btn: Embossed Capsule 7; fp-font-size: 14; fp-bgcolor: #C0C0C0; fp-transparent: 1" fp-title="Download" -->
Assignment 8 ERD

<img id="img12" alt="Download" height="27"
onmousedown="FP_swapImg(1,0,/*id*/'img12',/*url*/'button77.gif')" onmouseout="FP_swapImg(0,0,/*id*/'img12',/*url*/'button95.gif')"
onmouseover="FP_swapImg(1,0,/*id*/'img12',/*url*/'button76.gif')" onmouseup="FP_swapImg(0,0,/*id*/'img12',/*url*/'button76.gif')"
src="button95.gif" style="border: 0" width="135" /><!-- MSComment="ibutton" fp-style="fp-btn: Embossed Capsule 7; fp-font-size: 14; fp-bgcolor: #C0C0C0; fp-transparent: 1" fp-title="Download" -->

Assignment 8 SQL code (Queries)

<img id="img13" alt="Download" height="27"
onmousedown="FP_swapImg(1,0,/*id*/'img13',/*url*/'button79.gif')" onmouseout="FP_swapImg(0,0,/*id*/'img13',/*url*/'button96.gif')"
onmouseover="FP_swapImg(1,0,/*id*/'img13',/*url*/'button78.gif')" onmouseup="FP_swapImg(0,0,/*id*/'img13',/*url*/'button78.gif')"
src="button96.gif" style="border: 0" width="135" /><!-- MSComment="ibutton" fp-style="fp-btn: Embossed Capsule 7; fp-font-size: 14; fp-bgcolor: #C0C0C0; fp-transparent: 1" fp-title="Download" -->

Star-Schema / Data warehouse code

<img id="img7" alt="Download" height="27"
onmousedown="FP_swapImg(1,0,/*id*/'img7',/*url*/'button82.gif')" onmouseout="FP_swapImg(0,0,/*id*/'img7',/*url*/'button97.gif')"
onmouseover="FP_swapImg(1,0,/*id*/'img7',/*url*/'button81.gif')" onmouseup="FP_swapImg(0,0,/*id*/'img7',/*url*/'button81.gif')"
src="button97.gif" style="border: 0" width="135" /><!-- MSComment="ibutton" fp-style="fp-btn: Embossed Capsule 7; fp-font-size: 14; fp-bgcolor: #C0C0C0; fp-transparent: 1" fp-title="Download" -->

DOS
Batch file for CIS 350

<img id="img14" alt="Download" height="27"
onmousedown="FP_swapImg(1,0,/*id*/'img14',/*url*/'button7B.gif')" onmouseout="FP_swapImg(0,0,/*id*/'img14',/*url*/'button98.gif')"
onmouseover="FP_swapImg(1,0,/*id*/'img14',/*url*/'button7A.gif')" onmouseup="FP_swapImg(0,0,/*id*/'img14',/*url*/'button7A.gif')"
src="button98.gif" style="border: 0" width="135" /><!-- MSComment="ibutton" fp-style="fp-btn: Embossed Capsule 7; fp-font-size: 14; fp-bgcolor: #C0C0C0; fp-transparent: 1" fp-title="Download" --></td>
</tr>
<tr>
<td style="width: 299px" class="auto-style47">

Case Reports
individual
Case Report: The Appex Corporation

<img id="img27" alt="Download" height="27"
onmousedown="FP_swapImg(1,0,/*id*/'img27',/*url*/'button71.gif')" onmouseout="FP_swapImg(0,0,/*id*/'img27',/*url*/'button99.gif')"
onmouseover="FP_swapImg(1,0,/*id*/'img27',/*url*/'button70.gif')" onmouseup="FP_swapImg(0,0,/*id*/'img27',/*url*/'button70.gif')"
src="button99.gif" style="border: 0" width="135" /><!-- MSComment="ibutton" fp-style="fp-btn: Embossed Capsule 7; fp-font-size: 14; fp-bgcolor: #C0C0C0; fp-transparent: 1" fp-title="Download" -->
Case Report:

incident at Waco
Manufacturing

<img id="img28" alt="Download" height="27"
onmousedown="FP_swapImg(1,0,/*id*/'img28',/*url*/'button73.gif')" onmouseout="FP_swapImg(0,0,/*id*/'img28',/*url*/'button100.gif')"
onmouseover="FP_swapImg(1,0,/*id*/'img28',/*url*/'button72.gif')" onmouseup="FP_swapImg(0,0,/*id*/'img28',/*url*/'button72.gif')"
src="button100.gif" style="border: 0" width="135" /><!-- MSComment="ibutton" fp-style="fp-btn: Embossed Capsule 7; fp-font-size: 14; fp-bgcolor: #C0C0C0; fp-transparent: 1" fp-title="Download" -->
Case Report:

Connor Metals

<img id="img29" alt="Download" height="27"
onmousedown="FP_swapImg(1,0,/*id*/'img29',/*url*/'button75.gif')" onmouseout="FP_swapImg(0,0,/*id*/'img29',/*url*/'button101.gif')"
onmouseover="FP_swapImg(1,0,/*id*/'img29',/*url*/'button74.gif')" onmouseup="FP_swapImg(0,0,/*id*/'img29',/*url*/'button74.gif')"

src="button101.gif" style="border: 0" width="135" /><!-- MSComment="ibutton" fp-style="fp-btn: Embossed Capsule 7; fp-font-size: 14; fp-bgcolor: #C0C0C0; fp-transparent: 1" fp-title="Download" -->
group
(group) Case Report: The Topper Agency

<img id="img33" alt="Download" height="27"
onmousedown="FP_swapImg(1,0,/*id*/'img33',/*url*/'button40.gif')" onmouseout="FP_swapImg(0,0,/*id*/'img33',/*url*/'button3E.gif')"
onmouseover="FP_swapImg(1,0,/*id*/'img33',/*url*/'button3F.gif')" onmouseup="FP_swapImg(0,0,/*id*/'img33',/*url*/'button3F.gif')"
src="button3E.gif" style="border: 0" width="135" /><!-- MSComment="ibutton" fp-style="fp-btn: Embossed Capsule 7; fp-font-size: 14; fp-bgcolor: #C0C0C0; fp-transparent: 1" fp-title="Download" -->

(group) Case Report: A Harmless Prank

<img id="img34" alt="Download" height="27"
onmousedown="FP_swapImg(1,0,/*id*/'img34',/*url*/'button43.gif')" onmouseout="FP_swapImg(0,0,/*id*/'img34',/*url*/'button41.gif')"
onmouseover="FP_swapImg(1,0,/*id*/'img34',/*url*/'button42.gif')" onmouseup="FP_swapImg(0,0,/*id*/'img34',/*url*/'button42.gif')"
src="button41.gif" style="border: 0" width="135" /><!-- MSComment="ibutton" fp-style="fp-btn: Embossed Capsule 7; fp-font-size: 14; fp-bgcolor: #C0C0C0; fp-transparent: 1" fp-title="Download" -->
</td>
<td style="width: 313px" class="auto-style47">
HTML

(from this site)
Front page in HTML

<img id="img42" alt="Download" height="27"
onmousedown="FP_swapImg(1,0,/*id*/'img42',/*url*/'buttonEC.gif')" onmouseout="FP_swapImg(0,0,/*id*/'img42',/*url*/'buttonEA.gif')"
onmouseover="FP_swapImg(1,0,/*id*/'img42',/*url*/'buttonEB.gif')" onmouseup="FP_swapImg(0,0,/*id*/'img42',/*url*/'buttonEB.gif')"
src="buttonEA.gif" style="border: 0" width="135" /><!-- MSComment="ibutton" fp-style="fp-btn: Embossed Capsule 7; fp-font-size: 14; fp-bgcolor: #C0C0C0; fp-transparent: 1" fp-title="Download" -->
PHP
(from this site)

This page in PHP

<img id="img46" alt="Download" height="27"
onmousedown="FP_swapImg(1,0,/*id*/'img46',/*url*/'buttonC1.gif')" onmouseout="FP_swapImg(0,0,/*id*/'img46',/*url*/'button110.gif')"
onmouseover="FP_swapImg(1,0,/*id*/'img46',/*url*/'button109.gif')" onmouseup="FP_swapImg(0,0,/*id*/'img46',/*url*/'button109.gif')"
src="button110.gif" style="border: 0" width="135" /><!-- MSComment="ibutton" fp-style="fp-btn: Embossed Capsule 7; fp-font-size: 14; fp-bgcolor: #C0C0C0; fp-transparent: 1" fp-title="Download" -->
General Writing:
Looking Back in Time

<img id="img45" alt="View" height="27"
onmousedown="FP_swapImg(1,0,/*id*/'img45',/*url*/'button107.gif')" onmouseout="FP_swapImg(0,0,/*id*/'img45',/*url*/'button108.gif')"
onmouseover="FP_swapImg(1,0,/*id*/'img45',/*url*/'button106.gif')" onmouseup="FP_swapImg(0,0,/*id*/'img45',/*url*/'button106.gif')"
src="button108.gif" style="border: 0" width="135" fp-style="fp-btn: Embossed Capsule 7; fp-font-size: 14; fp-bgcolor: #C0C0C0; fp-transparent: 1" fp-title="View" />
Multiplying Fiction

<img id="img44" alt="View" height="27"
onmousedown="FP_swapImg(1,0,/*id*/'img44',/*url*/'button104.gif')" onmouseout="FP_swapImg(0,0,/*id*/'img44',/*url*/'button105.gif')"
onmouseover="FP_swapImg(1,0,/*id*/'img44',/*url*/'button103.gif')" onmouseup="FP_swapImg(0,0,/*id*/'img44',/*url*/'button103.gif')"
src="button105.gif" style="border: 0" width="135" fp-style="fp-btn: Embossed Capsule 7; fp-font-size: 14; fp-bgcolor: #C0C0C0; fp-transparent: 1" fp-title="View" />
Television and the route to
beautiful

<img id="img43" alt="View" height="27"
onmousedown="FP_swapImg(1,0,/*id*/'img43',/*url*/'button12.gif')" onmouseout="FP_swapImg(0,0,/*id*/'img43',/*url*/'button102.gif')"
onmouseover="FP_swapImg(1,0,/*id*/'img43',/*url*/'button11.gif')" onmouseup="FP_swapImg(0,0,/*id*/'img43',/*url*/'button11.gif')"
src="button102.gif" style="border: 0" width="135" fp-style="fp-btn: Embossed Capsule 7; fp-font-size: 14; fp-bgcolor: #C0C0C0; fp-transparent: 1" fp-title="View" />

```
                                                    </td>
                                                </tr>
</table>
</body> <a href="http://www.cdradyproductions.com?logout=1">Logout</a>
</body>

</html>
```

HTML code from cdradyproductions.com

The front page at cdradyproductions.com

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">

<head>
<meta content="text/html; charset=utf-8" http-equiv="Content-Type" />
<title>Charles Rady</title>
<style type="text/css">
.auto-style16 {
                                                    font-size: xx-large;
                                                    margin-left: 40px;
                                                    text-align: center;
}
.auto-style12 {
                                                    color: #000000;
}
</style>
<meta content="Charles Rady, Charley Rady" name="keywords" />
<meta content="Charles Rady" name="description" />
<script type="text/javascript">
<!--
function FP_preloadImgs() {//v1.0
 var d=document,a=arguments; if(!d.FP_imgs) d.FP_imgs=new Array();
 for(var i=0; i<a.length; i++) { d.FP_imgs[i]=new Image; d.FP_imgs[i].src=a[i]; }
}

function FP_swapImg() {//v1.0
 var doc=document,args=arguments,elm,n; doc.$imgSwaps=new Array(); for(n=2; n<args.length;
 n+=2) { elm=FP_getObjectByID(args[n]); if(elm) { doc.$imgSwaps[doc.$imgSwaps.length]=elm;
 elm.$src=elm.src; elm.src=args[n+1]; } }
}

function FP_getObjectByID(id,o) {//v1.0
 var c,el,els,f,m,n; if(!o)o=document; if(o.getElementById) el=o.getElementById(id);
 else if(o.layers) c=o.layers; else if(o.all) el=o.all[id]; if(el) return el;
 if(o.id==id || o.name==id) return o; if(o.childNodes) c=o.childNodes; if(c)
 for(n=0; n<c.length; n++) { el=FP_getObjectByID(id,c[n]); if(el) return el; }
 f=o.forms; if(f) for(n=0; n<f.length; n++) { els=f[n].elements;
 for(m=0; m<els.length; m++){ el=FP_getObjectByID(id,els[n]); if(el) return el; } }
 return null;
}
// -->
</script>
</head>
```

```
<body
onload="FP_preloadImgs(/*url*/'button5.jpg',/*url*/'button10.jpg',/*url*/'button11.jpg',/*url*/'button27.jpg',/*url*/'buttonE1.jpg',/*url*/'button
F.jpg',/*url*/'button28.jpg',/*url*/'button29.jpg',/*url*/'buttonE2.jpg',/*url*/'buttonF1.jpg')" style="background-color: #7F7F7F">

<table style="width: 100%">
                                                <tr>
                                                    <td style="width: 165px; height: 667px;">
                                                <p class="auto-style16" style="width: 186px">
                                                <span style="color: rgb(153, 153, 153);"><small><a href="audio">
                                                <img id="img1" alt="MUSIC" height="35"
onmousedown="FP_swapImg(1,0,/*id*/'img1',/*url*/'button10.jpg')" onmouseout="FP_swapImg(0,0,/*id*/'img1',/*url*/'button4.jpg')"
onmouseover="FP_swapImg(1,0,/*id*/'img1',/*url*/'button5.jpg')" onmouseup="FP_swapImg(0,0,/*id*/'img1',/*url*/'button5.jpg')"
src="button4.jpg" style="border: 0" width="175" class="auto-style12"><!-- MSComment="ibutton" fp-style="fp-btn: Embossed Capsule 5;
fp-font: Staccato222 BT; fp-font-size: 28; fp-bgcolor: #808080" fp-title="MUSIC" --></a></small></span></p>
                                                 <p class="auto-style16" style="width: 187px">
                                                <span style="color: rgb(153, 153, 153);"><small><small><a href="cis">
                                                      <img id="img2" alt="CIS" height="35"
onmousedown="FP_swapImg(1,0,/*id*/'img2',/*url*/'button27.jpg')" onmouseout="FP_swapImg(0,0,/*id*/'img2',/*url*/'button30.jpg')"
onmouseover="FP_swapImg(1,0,/*id*/'img2',/*url*/'button11.jpg')" onmouseup="FP_swapImg(0,0,/*id*/'img2',/*url*/'button11.jpg')"
src="button30.jpg" style="border: 0" width="175"><!-- MSComment="ibutton" fp-style="fp-btn: Embossed Capsule 5; fp-font:
Staccato222 BT; fp-font-size: 28; fp-bgcolor: #808080" fp-title="CIS" --></a></small></small></span></p>
                                                 <p class="auto-style16" style="width: 187px">
                                                <span style="color: rgb(153, 153, 153);"><small><small><a href="photo">
                                                   <img id="img4" alt="PHOTOS" height="35"
onmousedown="FP_swapImg(1,0,/*id*/'img4',/*url*/'buttonF.jpg')" onmouseout="FP_swapImg(0,0,/*id*/'img4',/*url*/'buttonD1.jpg')"
onmouseover="FP_swapImg(1,0,/*id*/'img4',/*url*/'buttonE1.jpg')" onmouseup="FP_swapImg(0,0,/*id*/'img4',/*url*/'buttonE1.jpg')"
src="buttonD1.jpg" style="border: 0" width="175"><!-- MSComment="ibutton" fp-style="fp-btn: Embossed Capsule 5; fp-font:
Staccato222 BT; fp-font-size: 28; fp-bgcolor: #808080" fp-title="PHOTOS" --></a></small></small></span></p>
                                                <p class="auto-style16" style="width: 184px; height: 37px;">
                                                 <span style="color: rgb(153, 153, 153);"><small><small><a href="video"><img
id="img5" alt="VIDEO" height="35" onmousedown="FP_swapImg(1,0,/*id*/'img5',/*url*/'button29.jpg')"
onmouseout="FP_swapImg(0,0,/*id*/'img5',/*url*/'button31.jpg')" onmouseover="FP_swapImg(1,0,/*id*/'img5',/*url*/'button28.jpg')"
onmouseup="FP_swapImg(0,0,/*id*/'img5',/*url*/'button28.jpg')" src="button31.jpg" style="border: 0" width="175"><!--
MSComment="ibutton" fp-style="fp-btn: Embossed Capsule 5; fp-font: Staccato222 BT; fp-font-size: 28; fp-bgcolor: #808080" fp-
title="VIDEO" --></a></small></small></span></p>
                                                 <p class="auto-style16" style="width: 186px">
                                                <span style="color: rgb(153, 153, 153);"><small><small>
                                                            <a href="articles">
                                                      <img id="img6" alt="Site" height="35"
onmousedown="FP_swapImg(1,0,/*id*/'img6',/*url*/'buttonF1.jpg')" onmouseout="FP_swapImg(0,0,/*id*/'img6',/*url*/'buttonD2.jpg')"
onmouseover="FP_swapImg(1,0,/*id*/'img6',/*url*/'buttonE2.jpg')" onmouseup="FP_swapImg(0,0,/*id*/'img6',/*url*/'buttonE2.jpg')"
src="buttonD2.jpg" style="border: 0" width="175"><!-- MSComment="ibutton" fp-style="fp-btn: Embossed Capsule 5; fp-font:
Staccato222 BT; fp-font-size: 28; fp-bgcolor: #808080" fp-title="Site" --></a></small></small></span></p>
                                                            </td>
                                                <td style="height: 667px"><span style="color: rgb(153, 153, 153);">
                                                <img alt="" height="699" src="cover2.jpg" width="992" /></span></td>
                                                </tr>
                                                <tr>
                                                      <td style="width: 165px"> </td>
                                                            <td> </td>
                                                </tr>
</table>

</body>

</html>
```

Technical Writing

Charles Rady
E. Pullen
Database Security
25 November 2011

AT&T's ICC ID headache:
The Goatse iPad incident

SUMMARY

A security breach was uncovered on AT&T's 3G network shortly after the release of Apple's coveted 3G-capable iPad in the summer of 2010. The breach occurred as a result of poor security planning (according to industry experts) on AT&T's 3G network. This breach allowed subscriber's ICC ID's and the e-mail addresses associated with those ID's to be easily downloaded. This has affected up to 114,000 subscribers including some high-profile names from the government and large corporations (see "who's affected" section for more). This breach was performed by two members of a group called "Goatse Security," a site that specializes in finding security holes and supposedly bringing those issues to the attention of the vendors in question. Their site is named after a famous shock site and more specifically a famous shocking picture on that site. In the past Goatse has revealed holes in web browsers such as Mozilla's Firefox and Apple's Safari along with some possible holes in some of Amazon's software.

Although Goatse claims to be working on the side of vendors, in this case an early release of the discovery of the breach by Goatse to Gawker Media (which promptly posted an article before AT&T even knew about the issue) caused their motivations to be brought into question. The delay in informing AT&T along with Goatse allegedly allowing third parties access to the hack that reveals the data caused an unknown number of others to also gain access. This act caught the attention of both the FCC and the FBI and when the smoke cleared two men, Andrew "Weev" Auernheimer and Daniel Spitler (both of Goatse Security) were arrested and charged with "conspiracy to access a computer without authorization and fraud in connection with personal information." (Ib Times) These men (according to authorities) clearly knew that they were breaking the law and even discussed destroying evidence, which the authorities claim shows that the men knowingly committed the illegal act. Whether this means they committed the act for criminal purposes is left up to the government to prove in court.

Apple is said to be a victim of AT&T as well in this case as it is not their fault (by design of the iPad operating system software) but instead entirely AT&T's fault. In looking at quotes from several members of the IT security industry (in the reactions from the industry section) you will see that most experts agree that AT&T was lax in their security preparations. It is also unclear if AT&T notified Apple of the incident when it did finally receive the information, which is in itself another question that should be aimed at AT&T and how they handle such breaches.

AT&T claims to have corrected the flaw in their code which allowed the unsophisticated hack to reveal an incredible number of high-profile e-mails on June 9th, 2010. It does not appear that AT&T appreciates the efforts of Goatse as arrests were made and charges filed. It should be noted that many companies do provide this type of service in good faith and work with large corporations and government agencies to improve security. It is apparent that not everyone in this business shares a high moral obligation to do the right thing and this incident is a good example of how easily one small immoral act by one or two people in an otherwise decent company can cause immense damage to an incredible number of people.

The Hack itself

The hack that revealed all of this private information was accomplished using what Chenxi Wang (an analyst at Forrester Research) calls a "parameter traversal attack" which is similar to the more common "directory

traversal attack" in which the goal of the attack is to cause a program to access a computer file that is not intended to be accessible. With such an attack the information is accessed in such a way that the original program does not act outside of its normal parameters. It does not "realize" that it is being tricked into passing unintended information.

The two men in question wrote a simple script in PHP that repeatedly attempted to pass an ICC ID to AT&T until it uncovered authentic ID's (automated data harvesting). Once the authentic ID's were discovered they were stored and eventually up to 114,000 ID's were revealed. The ID's allow the hacker to match the ID to an email that AT&T sends back when presented with an authentic ID. This is done by sending web requests to AT&T with iPad style "user agent" headers. These headers normally identify browser types to websites and here they provide a similar service. If a web request is sent with a valid ID in the header, the software assumes that it is connected to the genuine iPad that the ID is associated with and returns the iPad owner's e-mail address. The script in AT&T's code which does this is available to the public, so anyone who obtained the hack could easily duplicate Goatse's actions and obtain the same information. They knew how to pattern a code with the correct parameters in their PHP script by looking at pictures on Flickr taken by gadget enthusiasts which show the settings screen on their iPads and within that, their ICC ID. Armed with the pattern used for the ID's the two men were easily able to write the script that created random ID's until it happened on real ones. In reality the ploy is quite similar to the fictional ploy in the movie "WarGames" in which Mathew Broderick "programs" his PC to dial random numbers in search of active modems. Whenever his (fictional) system found a valid number, it saved it to a file along with the name of the system at the other end of the phone number. Goatse's ploy is really quite similar although modernized in comparison. They used the same method of programmed repeated trial & error to get the ID numbers that in their case uncovered e-mail addresses. When Gawker received the story about the hack from Goatse they were able to authenticate user's emails by (with permission) sending their supposed ID codes (obtained from Goatse) and receiving back e-mails which were authenticated by their owners.

Who Got Hacked?

While many experts do not consider this hack to be very damaging, many contend that it does reveal a weakness in AT&T's philosophy when it comes to their security planning. When looking at some of the revealed e-mail addresses, the following list represents SOME of the very famous and powerful people and companies who were affected:

1. White House Chief of Staff Rahm Emmanuel
2. Diane Sawyer
3. NY Times Co. CEO Janet Robinson
4. Harvey Weinstein
5. Michael Bloomberg
6. Many DARPA addresses (military)
7. William Eldridge (in charge of the largest operational Air Force B1 group)
8. Dow Jones
9. Conde Nast
10. Viacom
11. Time Warner
12. News Corp
13. HBO
14. Hearst,
15. U.S. Senate addresses
16. U.S. House of Representatives addresses
17. Department of Justice addresses
18. The FAA

19. The FCC
20. The National Institute of Health

And more…..

Below is a picture taken from Gawker showing a small part of the list:

IMAGE: Business Insider

You can see there from the ".gov" and ".mil" domains that some of these addresses belong to some very important people. With that in mind perhaps this attack was not as harmless as many contend. No one knows exactly how many people have this information or how many people on the AT&T network were affected. Although PROBABLY nothing major will come from it there is never a guarantee when such important people and organizations are affected.

REACTIONS FROM INDUSTRY EXPERTS

When asked about the AT&T ipad breach, several industry experts had plenty to say. While some of them speculate that the information obtained could later be used to spoof devices on the network or perform other malicious acts, others contend that the most harm that can come from this is phishing and spam e-mails sent to and from the captured addresses. One expert that does not give much credit to the attack's possible consequenses is NOKIA's Emmanuel Gadaix, who says "ICC IDs have not been involved in any breaches of the GSM network aside from this one." (business insider) Of course, this one is still relatively new so it does remain to be seen. Hacker and University of Virginia PhD Karsten Nohl agrees, saying "text and voice security is weak but data connections are typically well encrypted, so disclosure of ICC IDs has no direct security consequences." (business insider)They don't unless you consider the consequences that are already evident in the very case he was being asked about.

Most experts do take the attack a little more seriously and question AT&T's attention to detail when it comes to the security of their web applications. Karsten Nohl, who was quoted above taking the attack lightly does not take AT&T's stance on the issue the same way. When talking about AT&T he says "it's horrendous how customer data, specifically email addresses , are negligently leaked by a large telco provider." (business insider) Chenxi Wang, a security and risk management analyst at Forrester Research says about AT&T "it's the integrity of their application, if they had done a better job securing and testing their application, then this wouldn't have happened." (Wireless Week) She also mentions that she believes AT&T is being "cavalier with

security." (Wireless Week) Josh Phillip's, a senior malware researcher at Kapersky Lab says although the exposure is "not very serious," (Wireless Week) it raises concerns about AT&T's security. He goes on to say "I think that the more serious issue,is that based on this leak of information, AT&T most likely does not have a security team reviewing their customer-facing web apps prior to deployment – that is a much bigger issue," (Wireless Week) another vote for better security standards at AT&T.

Other experts had much the same sentiments regarding the breach. Jamz Yaneza, a threat researcher at Trend Micro says "ICC ID is just a security number and that information by itself isn't enough to do much; you have to put it together with a couple of different things to make it work." (Wireless Week) He also says that the main issue with exposed e-mail addresses is that it opens up users to be targets for spam and phishing. He points out that many American websites "you log in with your e-mail address and a random password," and "your e-mail is really rich… users should be concerned but I give kudos to AT&T for being able to patch this particular problem." (Wireless Week) Charlie Miller agrees with most of the others in saying that he doesn't believe the breach to be that serious but also agrees that "it does raise questions about AT&T's web applications and the security of them." (Wireless Week)

The FBI Probe

On June 11, 2010, the FBI initiated an investigation of this breach of security. The spokeswoman for the FBI Lindsay Godwin was quoted as saying "the FBI is aware of these possible computer intrusions and has opened an investigation to address the potential cyber threat." (Business Week) The FBI contacted Gawker Media, which had posted the story after being the first to receive it from Goatse Security and asked them to keep any documents related to the breach.

Although Goatse had claimed that they had informed AT&T before going public, the FBI found that this was not the case. Two members of the security firm had leaked the information early and had even had instant messenger conversations discussing the legal ramifications of their act, even discussing whether or not they should destroy evidence. Referring to "destroying evidence" and the acknowledgement of possible legal ramifications proved to the FBI that these men knew they were committing a crime and committed it anyway. Consequently both men were arrested and charged with conspiracy to access a computer without authorization and fraud in connection with personal information.

The men in question are Andrew "Weev" Auernheimer and Daniel Spitler, both of Goatse Security. Although they assumed that they could mostly face civil problems, the charges aimed at them are federal which could mean prison time for both men. The government presented captured instant messenger communications between the two men and some of them clearly show knowledge of their hack to be a criminal act. After Spitler spent some time explaining the benefits of mining e-mail addresses to Auernheimer, he went on to say "dunno how legal this is or if they could sue for damages. Auernheimer responded with "absolutely may be legal risk yeah, mostly civil you absolutely could get sued to f---." It was after they released the story to Gawker that they discussed possibly destroying evidence.

AT&T claims that the incident cost them $73,000 which was used to quickly patch the security flaw. In the government's complaint it is stated that Auernheimer claimed credit for the hack, or at least publicizing it. The government still has to make the case that these men did this for criminal purposes, as they still content that their initial intent was to reveal weaknesses to AT&T so that they could patch them before they were exploited. Their conversations do not lend credence to this but that is for a court to decide.

AT&T's Response to the Public

AT&T sent a letter to customers to apologize to them for the breach and reassure them that they can continue to use their iPads with confidence. An actual portion of the letter is presented below as I myself am an early adopter of the iPad, continue to be an AT&T subscriber and iPad user, and myself received a copy of the letter.

See next page for an excerpt from the actual letter

-Excerpt from the AT&T letter to consumers

"On June 7 we learned that unauthorized computer 'hackers' maliciously exploited a function designed to make your iPad log-in process faster by pre-populating an AT&T authentication page with the email address you used to register your iPad for 3G service. The self-described hackers wrote software code to randomly generate numbers that mimicked serial numbers of the AT&T SIM card for iPad – called the integrated circuit card identification (ICC-ID) – and repeatedly queried an AT&T web address. When a number generated by the hackers matched an actual ICC-ID, the authentication page log-in screen was returned to the hackers with the email address associated with the ICC-ID already populated on the log-in screen.

The hackers deliberately went to great efforts with a random program to extract possible ICC-IDs and capture customer email addresses. They then put together a list of these emails and distributed it for their own publicity.

As soon as we became aware of this situation, we took swift action to prevent any further unauthorized exposure of customer email addresses. Within hours, AT&T disabled the mechanism that automatically populated the email address. Now, the authentication page log-in screen requires the user to enter both their email address and their password."

Works Cited

Tate, Ryan. “Apple’s Worst Security Breach: 114,000 iPad Owner’s Exposed.” *Businessinsider.com.* 2010-6: 7, 2011-2 <http://www.businessinsider.com/apples-worst-security-breach-114000-ipad-owners-exposed-2010-6>

Rolvella, David E. “FBI Probes AT&T Network’s iPad E-Mail Security Breach (Update2)” *Businessweek.com.* 2010-6: 3, 2011-2 <http://www.businessweek.com/news/2010-06-11/fbi-probes-at-t-network-s-ipad-e-mail-security-breach-update2-.html>

Ramsay, Maisie. “iPad Security Breach Not Serious But Easily Avoidable.” *Wirelessweek.com*. 2010-6: 3, 2011-2 http://www.wirelessweek.com/news/2010/06/Devices-iPad-Security-Breach-Avoidable-Computers/

Unknown. “Gawker iPad Security Breach.” *Mashable.com*.2010-6: 3,2011-2 http://mashable.com/2010/06/09/att-gawker-ipad-security-breach

Unknown. “Hackers Charges in iPad Security Breach.” *Ibtimes.com.* 2010-6:4,2011-2 http://www.ibtimes.com/articles/102591/20110119/hackers-charged-in-ipad-security-breach.htm

AT&T Letter, author unknown. “*AT&T letter to customers*.” *AT&T.* 2010: 1,2011-2. Sent to all customers with 3G iPad accounts.

*NOTE: images come from *businessinsider,* however; they give credit to GAWKER for the images.

Standards and Practices for Charitable Donations

The business practices surrounding non-profit organizations have always seemed to go unnoticed by regulators. Until fairly recently there were few (or no) regulations surrounding how a charitable organization handled it's donations or tracked its donors. This is beginning to change for both the charitable organizations as well as the donors themselves. In this short paper I will address the rules surrounding making a donation to a non-profit and take a quick look at a few leading providers of donor tracking software packages.

In the past it seems that the IRS looked upon charitable donations with a trusting eye. It's almost as if they and their taxpayers were operating on the buddy system. For the donor, all that was required to be able to donate and then use it as a tax deduction was to simply make a donation and keep the check. In the case of non-cash donations they could pretty much get away with writing their own receipts in most cases. In fact many non-profits have even made it popular to simply give donors blank receipts automatically, this allows them to value their own items for the purposes of tax deductions and makes for possible easy tax evasion!

This has all been changing over the past few years. As of when this paper was written, any donation over $250 must not only have the canceled check for proof but also written verification from the charitable organization that received the money. The IRS has been stepping up its audits on non-profits and taking a closer look at deducted donations for taxpayers (an unusual thing until relatively recent times). Proper documentation is also required for any non-cash donations, and obtaining proof of how much was donated as well as how much of that donation is tax deductible is becoming more confusing as regulations thicken. I will elaborate more on how to properly deduct non-cash donations a little further down in the paper.

When it comes to non-cash donated items a good question to ask is this: who is legally responsible for correctly valuing the item donated and ensuring that it is documented correctly? The answer may surprise you. The responsibility falls squarely on the shoulders of the donors themselves, not the charitable organizations. Perhaps this is why so many are willing to dole out blank receipts for donated items. This is something donors need to know before they try to make deductions based on their charitable contributions.

Another regulation that may surprise and is very important when tax time arrives is the percentage of non-cash items that can be donated. If you donate an item and choose a fair market value for said item, not only can your chosen price come under scrutiny but also how much of the claimed value was deducted can also be scrutinized. This is because is some cases (such as items won bidding in a charity auction) you cannot deduct the donation at all. In some other cases when the donor receives something (for instance when you receive something in return for the item, even if it's only a soup kitchen lunch) you are required to subtract the value of what you received from the value of what you donated). In cases involving large items such as cars, the organization must use the donated vehicle or large item for you to fully qualify. If the organization sells the donated item then the amount they received is subtracted from the value you (the donor) has written (not very fair, is it?)

The category that the IRS places your donation in can also have an effect on its ability to qualify as a tax deduction. Was it politically motivated? Is it in response to a local fund raiser for a challenged family (for instance one who lost their home in a flood or was robbed and has a fund set up locally for them)? Are you deducting a cash value for time spent volunteering? Did you donate to a church (without documentation)? You can't deduct any of these things at all, period. You need to know what qualifies for tax write-offs and what doesn't before you start writing up your next tax return. Non-profits also need to be savvy so that they can properly advise their constituents when faced with decisions on what to donate and how.

This all sounds like its getting pretty crowded with regulations, but wait, there's more! Not only must you be careful not to tread on the IRS' rules when you go about making your yearly tax deductions, you must also fill out some paperwork in the case of non-cash donations valued at over $500 at the time of donation and you must go as far as to get a professional appraisal for items valued at over $5,000. In the case of items

between $500 and $4,999 donors are required to fill out form 8283 with the IRS which can be obtained on the IRS website.

What does this have to do with online donor tracking software providers? Well, in the case of non-cash items not too much as most of these items will be donated physically at the charity's brick & mortar locations. The responsibility of handling these donations correctly will fall on the shoulders of the volunteers and employees of the organization, and not their donor tracking software. The online providers mostly track the donors information and allow for online money donations. In the case of such cash donations, the sites (all of the sites I have perused so far and I'd be willing to bet all of the sites out there period) will provide adequate proof of the donation and canceled checks can be obtained from the bank that they were drawn on. These types of donations are safe bets for deductions because they are handled in such a way as there can be no argument over how much a donation was and when it was made. These online services do a great job of keeping track of money donations.

In the case of gifts-in-kind (non-cash donations) most of the online providers offer very little in the way of collecting them (as would be obvious since they are often cloud based, and it's hard to send your old refrigerator through a virtual cloud!) Although sending some gift-in-kind items through the mail works, it is most often best to do such donations in person. Most of these businesses will offer services that promote gifts-in-kind and offer some way of helping businesses to attract them, they do not offer much in the way of properly valuing and documenting them. These activities are best performed by the organizations themselves as they often require hands-on attention to properly receive them. Hand in Hand ministries has approached us with the hope of finding a solution that will display on their web-page needed items and the quantities needed. Although this is a very useful tool it has no connection to valuing or documenting the items and therefore it is not really relevant to what I am talking about here.

As the responsibility for correctly valuing and documenting gift-in-kind items falls squarely on the shoulders of the donor, and both the donors and the organizations are responsible for their respective tax decisions, any online donor management service should not be counted on for such activities. In most cases they will do little or nothing when it comes to enforcing or accommodating policies and procedures surrounding gift-in-kind, and the only accommodating they do on cash donations is simply keeping records of the donations for you. While this may seem an advantage, when you consider that any cash or credit transaction will have secure and complete records attached to the bank or other financial institution you use, this is in fact a case of redundancy, and not in any way an advantage provided by these services. As far as donor management services go, you should count on them to keep a database of your constituents and provide a place for announcements and online donations. As far as obeying the tax laws, that is one hundred percent the responsibility of the organizations and donors, online services aren't and should not be held responsible nor should you trust completely in any of their services that claim to fulfill such obligations. Paying your taxes and obeying the tax laws are your responsibility and your responsibility only.

Some non-profit donor management services

As I discussed in the previous section, I do not see how any of these services can have an impact on how a business decides to accommodate their obligations to the government. They all seem to mirror eTapestry and SalesForce pretty closely with their offerings with the most noticeable difference being the price that they charge for their services. I can say right off the bat that none of the services I have checked out are good enough for me to recommend them to Hand in Hand Ministries due to any ability on the part of the software to follow procedures or laws. There are, however; some abilities in these packages that I am reviewing here that could make them a possible third option next to eTapestry and SalesForce due to functionality such as built-in recurring donation handling and other very useful tools that may be easier to use than the competition. As far as policies of the non-profit pertaining to how they handle gifts-in-kind and cash donations, that is up to them and I don't see it being any easier or harder with any of these services. Most of

that is up to the organization and the services merely provide the database services that the non-profit is either unwilling or incapable of handling themselves.
With that said here is a very cursory glance at some other options:

DonorPerfect by SofterWare

DonorPerfect (makes me wonder if they used WordPerfect to write up their original proposal and got stuck on the name) is a software package that advertises "fully integrated programs" that can handle any type of echeck or credit card transaction. Interestingly, this company does offer (and advertises right on their front page) the automatic collection of pre-authorized recurring pledge payments. That line ought to make Hand in Hand sit up and take notice as that was one of their pet peeves with the eTapestry system. DonorPerfect is a fairly low-cost alternative to eTapestry but it is far from free, which still keeps NPSP at the top of my list for Hand in Hand. DonorPerfect offers three tiers of service, the express, essentials, and premium services. Unfortunately for Hand in Hand, their number of constituents being tracked exceeds the two cheaper versions and would force them to choose the premier package with this company. The premier package costs $199 per month plus support for between 5,001 and 25,000 records. This does include everything that they offer and some of it is very useful. Some examples of the services included with the premier service are:

1) CRM database
2) Donation processing
3) Reporting and analysis
4) Backups and "enhancements"
5) EZ-GIFT recurring gift system
6) Advanced data import tool
7) Interfaces to major accounting systems
8) SmartActions business rules
9) XML-Application programming interface

It would seem that this software provides some services wanted by Hand in Hand that is not offered by eTapestry. The most noticeable is the automatic recurring pledges, a service that I know Hand in Hand is very interested in. With the data migration tool being included this software package comes very close to becoming a candidate itself and I would place it under NPSP as a low-cost (but not free) alternative .

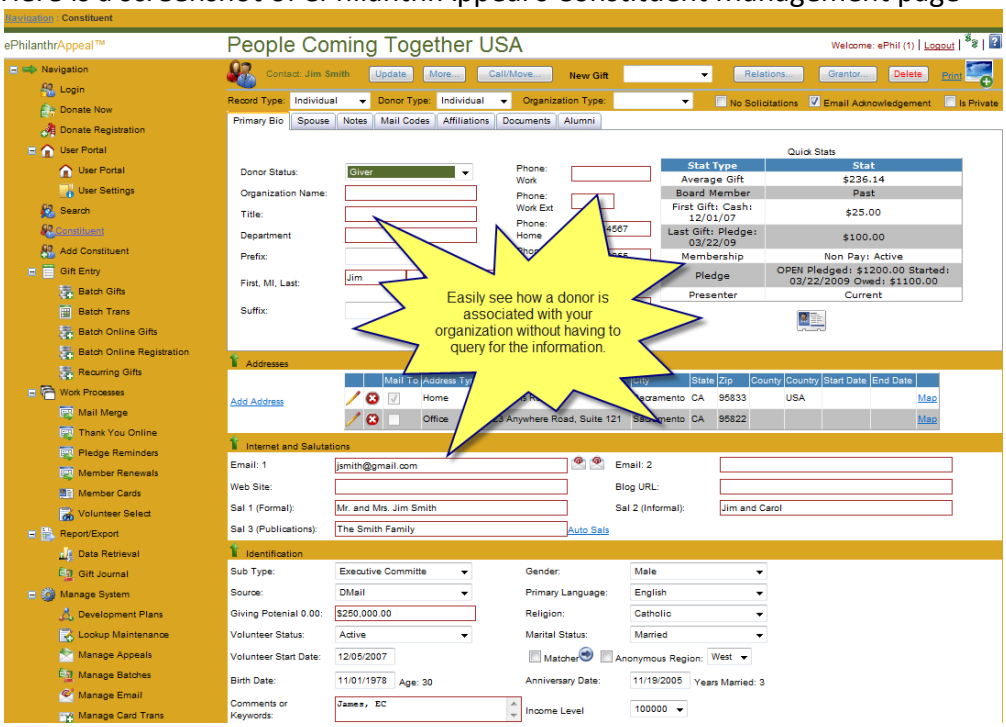
ePhilanthrappeal by FundTrack software

This oddly named software package touts itself as a great way to analyze your data. Their website is headed by the sentence "don't just collect your data *analyze* it with ePhilanthrAppeal." This package allows a business to choose between an entirely web-based experience or an installed version. The installed version would require a more planned out system approach than Hand in Hand is currently running (by this I mean they would need to be on the same type of system and not a collection of random Macs and PC's with widely varying systems). For this reason I would recommend that if considered at all, only the web-based version should be considered.
Some of the features included with this package are:

1) Easy user portal
2) Thank you process
3) Integrated Online Donations
4) Online Event Registrations
5) Reporting
6) eMail services
7) Multiple user roles

8) Credit card processing without monthly fees
9) Track single & multi payment pledges (recurring pledges)
10) Easy entry for in-kind gifts
11) Integration with QuickBooks and CYMA accounting

Here is a screenshot of ePhilanthrAppeal's Constituent Management page


ePhilanthrAppeal™
People Coming Together USA
Easily see how a donor is associated with your organization without having to query for the information.

Pricing for this software package is not readily available to curious web-surfers. To get an accurate rate you must request a price quote from their website (listed in works cited section). For this reason I cannot recommend this software to Hand in Hand. Although it does seem to offer many features that Hand in Hand is looking for, it is not forward enough with its pricing scheme nor does it top DonorPerfect with its offerings. ePhilanthrAppeal is simply another option to keep in mind, although at the bottom of the list.

Sage Non-Profit Solutions

Sage is a provider of many different accounting and CRM solutions for a wide range of businesses. They also offer a non-profit solution at a discount from their regular business services. Some of their success stories include the Girl Scouts of America and Child Care Action Campaign which have done remarkably well using sage nonprofit solutions software.

This software package seems to be more of an accounting package than it is a tailored package for constituent tracking and management. It can perform these of course, I just believe that the emphasis with them is not where it would need to be in order for them to be of any interest to Hand in Hand Ministries. Some of the features that they advertise are:

1) Targeted solicitations, communications, and acknowledgement
2) Easy donation reception and managing
3) Recurring pledge management
4) Plan, organize, and manage all aspects of a fundraiser
5) Built in "Help me' and "Show me" assistance

6) Flexible reporting
7) Fully customizable

As with ePhilanthrAppeal, Sage does not offer up-front payment information. They also require the business to "apply" for more information in order to get a realistic price. Because of this (and the repeating options available in all three) I would not recommend this software package to Hand in Hand.

After looking at these three software packages and reviewing the IRS laws pertaining to receiving and deducting donations, I have some to the conclusion that as far as the law is concerned, none of these add any functionality that should influence a decision. As far as helping to manage Gift-In-Kind and cash donations the only service that is completely lacking and needs to be off the list is eTapestry. All three of the packages I have looked at here can do the job and once again it is likely to come down to cost. The cheapest one that can do the job will probably be the winner. From these three choices I would only add DonorPerfect to our current list of prospects. This brings the current list to the following three options (besides eTap which is their current system)

1) Convio Common Ground
2) SalesForce NPSP
3) DonorPerfect

At this time, due to the free nature of NPSP, I consider it to be the current front-runner with Convio in second (due to its relationship with NPSP and the possibility of easily switching from one to the other) and DonorPerfect coming in last. The others we have looked at (including eTapestry) do not make the list due to either cost or functionality constraints.

Works Cited

Sage website. Sage Inc., 28 September 2010
< www.sagenonprofit.com>

DonorPerfect website. SofterWare Inc., 28 September 2010
< http://www.donorperfect.com/transaction-processing>

"Donation Management Software Tools" Capterra website, 28 September 2010
<www.Capterra.com/donation-management-software>

ePhilanthrAppeal website. FundTrack Software inc., 28 September 2010
<www.fundtracksoftware.com>

Hoelzer, Annette "How to Make Charitable Contributions without Violating IRS regulations" *Smart Business* April 2010
<sbonline.com/local/article/19439/65/1How_to_make_charitable_contributions_without_violating_IRS_regulations.aspx>

"Critical Issues in Financial Accounting Regulation for Nonprofit Organizations" website. 28 September 2010. <
www.muridae.com/nporegulation/accounting.html>

Case Report : Waco Manufacturing

Note: There are 5 complete reports available at cdradyproductions.com/cis/localarticle4

Background

The "Incident at Waco Manufacturing" is a fictitious case designed by professor John Sviokla of the Harvard Business School in 1989. The case is designed to explore the effects of an informated organization on its employees and management. The case describes a company (Waco) that installed a new information technology system that included placing transceivers in the walls of one of its plants. These transceivers (capable of two way communication) were placed approximately every 25 feet and interacted with badges worn by the employees. Even in 1981 (the setting of this case) with transceivers placed this closely together, an employee's movements could be monitored in real time. This allowed Waco to locate employees instantly to within a small distance of their actual location. Close enough for the system to route calls to the phone nearest the employee in question (remember, it is 1981 and there aren't any cell phones to be had).

Another use of the technology is storage of the data over time and then creating reports of a particular employee or group of employee's movements over a specified period of time. This kind of omnipotent management tool allows no hiding and makes visible activities that were once unable to be transformed into a black and white paper report, with no time omitted. This kind of "surveillance" is something that has never before been available to Waco or anybody else for that matter. The problem in this case arises when an area manager (Monique Saltz) informed an engineering manager (Monk Barber) that she was unhappy with the design of a new composite based product. She had made several attempts to meet with the engineers to discuss the problem and arrange a re-design but was unable to find satisfaction. She spoke to the plant manager (Shelly Tomaso) about the problem. Their solution included checking records produced by the new system on all 4 men in question. Their search of the records revealed that all four men had never even been in the same room together since early in the year.

Problem statement

The problem: Waco has introduced new technology that is having unforeseen effects. The system, which was installed to increase management's ability to locate employees quickly when needed and route phone calls to their locations (among other real-time duties) has been used to reveal an apparent lack of drive in the engineering department. Although on the surface this appears to be a case of some lazy unfocused employees unwilling to come together and do the work, nothing is ever mentioned about WHY they were not together. Were the man assigned to tasks that were involved and important and had good reason to be separate? The case only tells us how badly Monique wants these parts re-designed. Nowhere does it mention anything aside from her frustration. The system only shows where the men were, not what they were doing.

This raises a group of questions about what the management should do in this situation. Should the men be disciplined? That depends on what they were actually doing. Should the system be used in such an Orwellian manner and is this ethical? Yes, and yes. Any move to increase productivity is an attempt to make money and that is what a business is supposed to do.

Stakeholders

- Plant manager Shelly Tomasco
- Plant engineers
- Engineering manager Monk Barber
- The three engineers McCoy, Frank, and Gogan
- Area manager Monique Saltz

Porter's Five Forces

- **Threat of new entries:** Low. Waco is introduced to us as a "leading supplier of custom-machined parts to the automotive industry" (Cash) This would lead me to believe that they are a high-volume supplier which needs expensive custom machinery to do their business. Because of this and the limited number of auto manufacturers that could be courted by potential new-entries, the threat is low.
- **Threat of Substitutes:** High. Waco is not the only established company providing the auto makers. Another company with the ability could make the parts if provided with the specs from the auto manufacturer in question.
- **Supplier Power:** High. Waco cannot do business without the metals and composite materials provided to them without their suppliers. They could not operate without these materials therefore supplier power is high.
- **Buyer Power:** High. These are custom parts, if they are not ordered by the auto makers then they will be nearly impossible to sell.
- **Degree of Rivalry:** This is hard to say. Not enough information is given, however; I would assume that there is plenty of already-established companies competing with Waco so I would assume they would need to consider their competition – so I will assume that the degree of rivalry is high.

Courses of action

1) Do nothing. If the company does nothing, the employees may not become aware of management's ability to report on their every movement over a period of time. This could prevent a morale problem from occurring. The men in question are supposedly highly trained engineers, there is a very good chance that they actually have been working on things that ARE very important to Waco… just not all-powerfully important to Ms. Saltz. Nowhere in the case does it describe anything that they were doing except for what Ms. Saltz reports that they are NOT doing. The system also cannot tell if the men had discussed the problem, only that they had not been physically in the same room. They could have made phone calls. In short, this whole incident revolves around Ms. Saltz and her nervous breakdown - as she said herself, "I'm at wit's end." (Cash) Not enough information is given to determine whether or not discipline or a promotion is called for. To explain that: If the men had been avoiding the work for no good reason, then they should be disciplined, if they had discussed it yet decided to work on things they deemed (or Waco's higher management had deemed) to be more important than the parts Ms. Saltz is so worried about. Also,

what stopped her from using the system to locate the men, then walk down to their location and demand an answer.

2) Talk to the men carefully without revealing the reporting power of the system. If the management does this, they may discover why the men haven't answered Ms Saltz yet, what they were doing, and whether or not disciplining them is an option. With a few carefully chosen questions, they could find the reason for the delay and move the men onto the composite piece that is needed without creating a backlash from the employees over the surveillance. Because of the system, management would most likely be able to gauge the truth of the men's statements should they be suspected of being untruthful. Doing this should satisfy Ms. Saltz, the rest of management, the customer waiting on the parts, and the important and hard-to-replace engineers who Ms Saltz believes are not doing their jobs. To accuse them falsely (remember, nothing is said of what they were ACTUALLY doing) would certainly cause friction in this Waco plant which could lead to larger problems down the road.
3) Call the men out on their movements. Reveal the reporting system (most today would assume they could be tracked this way – not so in 1981, I believe it would come as a surprise and a shock to most at that time) then act on the information provided by the system without regard to statements made by the men about why the work wasn't done. Then discipline them. This could be a very bad course of action when it comes to employee morale. It would also be based completely on information that does not provide an all-seeing account. As amazing as this technology is for 1981, there are too many holes in it. I've already mentioned that they could have been calling each other, they also could have met elsewhere and discussed it, they could have actually met but left their badges in their offices (engineers might know the capabilities of the system). With all these holes, this course of action would most likely be seen as incredibly unfair and make relations between employees and management so bad that the system would have to be curtailed in order to restore morale.

Best course of action and why

The best course of action for Waco to take would be action number two. I believe this because this action maximizes the possible positive results, the work gets finished, the men stay happy, Ms Saltz becomes happy. It also minimizes the negative. I have pointed out that this technology is not a normal occurance in 1981. While we might not think it odd, someone in 1981 might not even believe it to be possible. To enlighten the employees on the power of the system to keep long-term tabs on them could be very detrimental.

Conclusion

The word "informate" has been used here to describe the effects (other than automation) that information technology has on our business practices. When a business process is automated, it "is meant to accomplish the same tasks as the human body" (Cash) replacing the human and allowing them a less intensive role, or no role at all. When a business is "informated," we are describing the effects that information technology has on a business as a whole. Employees are needed to bring intellectual skills to the table more than other skills, for instance, the Cash book states "instead of *dumber* people, we are discovering that people need whole new skill sets as the basis for competence and excellence in this new world." (Cash) This goes contrary to the story of the

old Chinese man at the opening of "Images of Organization" by Gareth Morgan who said "whoever uses machines does all his work like a machine….he who does his work like a machine grows a heart like a machine…"(Morgan)

Instead of a bunch of human machines, we are seeing a higher level of intellectual work being done. For instance, people need to understand a database to operate it. Automation is everywhere and this new-ish word brought to us by the Cash text "informate" is a little hard to define. In a sense it is still automating, we are simply automating the job of keeping track of our employees, and a computer system can do that better than a human. It was not something that wasn't attempted before with humans doing the job. In short, I believe that the term "informated" is a little unnecessary. I have decided to talk about it a little because it is used quite a bit in the related reading.

To conclude, I believe that Waco should tread carefully with this new technology. They should be cautious about the use of it and especially the possibility of it becoming public knowledge among all of the employees. I certainly believe number two is their best option, and I believe the word "informated" is unnecessary.

Works Cited

Cash, James I..”Corporate Information Systems Management” from CORPORATE INFORMATION SYSTEMS MANAGEMENT: THE CHALLENGES OF MANAGING IN AN INFORMATION AGE.,(McGraw-Hill companies, inc. –books 1999)/5 ed.

Morgan, Gareth Images of Organization. SAGE publications, CA, 1986

APPENDIX A: The Library Class Hierarchy
Library Class

```
// File: Library.cs
// This file creates a basic Library class that stores a list
// of LibraryItems and a list of LibraryPatrons. It allows items
// to be checked out by patrons. The lists are accessible to other
// classes in the same namespace (LibraryItems).

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace LibraryItems
{
    [Serializable]
    public class Library
    {
        // Namespace Accessible Data - Use with care
        internal List<LibraryItem> items;     // List of items stored in Library
        internal List<LibraryPatron> patrons; // List of patrons of Library

        // Precondition:  None
        // Postcondition: The library has been created and is empty (no books, no patrons)
        public Library()
        {
            items = new List<LibraryItem>();
            patrons = new List<LibraryPatron>();
        }

        // Precondition:  None
        // Postcondition: A patron has been created with the specified values for name and ID.
        //                The patron has been added to the Library.
        public void AddPatron(String name, String id)
        {
            patrons.Add(new LibraryPatron(name, id));
        }

        // Precondition:  theCopyrightYear >= 0 and theLoanPeriod >= 0
        // Postcondition: A library book has been created with the specified
        //                values for title, publisher, copyright year, loan period,
        //                call number, and author. The item is not checked out.
        //                The book has been added to the Library.
        public void AddLibraryBook(String theTitle, String thePublisher, int theCopyrightYear,
            int theLoanPeriod, String theCallNumber, String theAuthor)
        {
            items.Add(new LibraryBook(theTitle, thePublisher, theCopyrightYear, theLoanPeriod,
                theCallNumber, theAuthor));
        }

        // Precondition:  theCopyrightYear >= 0 and theLoanPeriod >= 0 and
        //                theMedium from { DVD, BLURAY, VHS } and theDuration >= 0
        // Postcondition: A library movie has been created with the specified
        //                values for title, publisher, copyright year, loan period,
        //                call number, duration, director, medium, and rating. The
        //                item is not checked out.
        //                The movie has been added to the Library.
```

```
public void AddLibraryMovie(String theTitle, String thePublisher, int theCopyrightYear,
    int theLoanPeriod, String theCallNumber, double theDuration, String theDirector,
    LibraryMediaItem.MediaType theMedium, LibraryMovie.MPAARatings theRating)
{
    items.Add(new LibraryMovie(theTitle, thePublisher, theCopyrightYear, theLoanPeriod,
        theCallNumber, theDuration, theDirector, theMedium, theRating));
}

// Precondition:  theCopyrightYear >= 0 and theLoanPeriod >= 0 and
//                theMedium from { CD, SACD, VINYL } and theDuration >= 0 and
//                theNumTracks >= 0
// Postcondition: A library music item has been created with the specified
//                values for title, publisher, copyright year, loan period,
//                call number, duration, director, medium, and rating. The
//                item is not checked out.
//                The music item has been added to the Library.
public void AddLibraryMusic(String theTitle, String thePublisher, int theCopyrightYear,
    int theLoanPeriod, String theCallNumber, double theDuration, String theArtist,
    LibraryMediaItem.MediaType theMedium, int theNumTracks)
{
    items.Add(new LibraryMusic(theTitle, thePublisher, theCopyrightYear,
    theLoanPeriod, theCallNumber, theDuration, theArtist,
    theMedium, theNumTracks));
}

// Precondition:  theCopyrightYear >= 0 and theLoanPeriod >= 0 and
//                theVolume >= 0 and theNumber >= 0
// Postcondition: A library journal has been created with the specified
//                values for title, publisher, copyright year, loan period,
//                call number, volume, number, discipline, and editor. The
//                item is not checked out.
//                The journal has been added to the Library.
public void AddLibraryJournal(String theTitle, String thePublisher, int theCopyrightYear,
    int theLoanPeriod, String theCallNumber, int theVolume, int theNumber,
    String theDiscipline, String theEditor)
{
    items.Add(new LibraryJournal(theTitle, thePublisher, theCopyrightYear,
    theLoanPeriod, theCallNumber, theVolume, theNumber,
    theDiscipline, theEditor));
}

// Precondition:  theCopyrightYear >= 0 and theLoanPeriod >= 0 and
//                theVolume >= 0 and theNumber >= 0
// Postcondition: A library magazine has been created with the specified
//                values for title, publisher, copyright year, loan period,
//                call number, volume, and number. The item is not checked out.
//                The magazine has been added to the Library.
public void AddLibraryMagazine(String theTitle, String thePublisher, int theCopyrightYear,
    int theLoanPeriod, String theCallNumber, int theVolume, int theNumber)
{
    items.Add(new LibraryMagazine(theTitle, thePublisher, theCopyrightYear,
    theLoanPeriod, theCallNumber, theVolume, theNumber));
}

// Precondition:  None
// Postcondition: The number of patrons in the library is returned
public int GetPatronCount()
{
    return patrons.Count;
}
```

```
// Precondition:  None
// Postcondition: The number of items in the library is returned
public int GetItemCount()
{
    return items.Count;
}

// Precondition:  0 <= itemIndex < GetItemCount()
//                0 <= patronIndex < GetPatronCount()
// Postcondition: The specified item will be checked out by
//                the specifed patron
public void CheckOut(int itemIndex, int patronIndex)
{
    if ((itemIndex >= 0) && (itemIndex < GetItemCount()) &&
        (patronIndex >= 0) && (patronIndex < GetPatronCount()))
        items[itemIndex].CheckOut(patrons[patronIndex]);
}

// Precondition:  0 <= bookIndex < GetItemCount()
// Postcondition: The specified book will be returned to shelf
public void ReturnToShelf(int itemIndex)
{
    if ((itemIndex >= 0) && (itemIndex < GetItemCount()))
        items[itemIndex].ReturnToShelf();
}

// Precondition:  None
// Postcondition: The number of items checked out from the library is returned
public int GetCheckedOutCount()
{
    int checkedOutCount = 0; // Running count of checked out books

    foreach (LibraryItem item in items)
        if (item.IsCheckedOut())
            ++checkedOutCount;

    return checkedOutCount;
}

// Namespace Helper Method - Use with care
// Precondition:  None
// Postcondition: The list of items stored in the library is returned
internal List<LibraryItem> GetItemsList()
{
    return items;
}

// Namespace Helper Method - Use with care
// Precondition:  None
// Postcondition: The list of patrons stored in the library is returned
internal List<LibraryPatron> GetPatronsList()
{
    return patrons;
}

// Precondition:  None
// Postcondition: A string is returned presenting the libary in a formatted report
public override string ToString()
{
    // Using StringBuilder to show use of a more efficient way than String concatenation
    StringBuilder report = new StringBuilder(); // Will hold report as being built
```

```
            report.Append("Library Report\n");
            report.Append(String.Format("Number of items stored:      {0,4:d}{1}", GetItemCount(),
System.Environment.NewLine));
            report.Append(String.Format("Number of items checked out: {0,4:d}{1}",
GetCheckedOutCount(), System.Environment.NewLine));
            report.Append(String.Format("Number of patrons stored:    {0,4:d}", GetPatronCount()));

            return report.ToString();
        }
    }
}
```

Library Item Class

```
// File: LibraryItem.cs
// This file creates an abstract LibraryItem class that will serve
// as the base class of a hierarchy of library items that keep
// track of common information and can be checked out by
// LibraryPatrons.
// LibraryItem HAS-A LibraryPatron (when the item is checked out)

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace LibraryItems
{
    [Serializable]
    public abstract class LibraryItem
    {
        private LibraryPatron patron;  // The person that has the book checked out (null otherwise)
        private bool itemCheckedOut;   // The item's checked out status
        private int itemCopyrightYear; // The item's year of copyright
        private int itemLoanPeriod;    // The item's loan period

        // Precondition:  theCopyrightYear >= 0 and theLoanPeriod >= 0
        // Postcondition: The library item has been initialized with the specified
        //                values for title, publisher, copyright year, loan period and
        //                call number. The item is not checked out.
        public LibraryItem(String theTitle, String thePublisher, int theCopyrightYear,
            int theLoanPeriod, String theCallNumber)
        {
            Title = theTitle;
            Publisher = thePublisher;
            CopyrightYear = theCopyrightYear;
            LoanPeriod = theLoanPeriod;
            CallNumber = theCallNumber;

            ReturnToShelf(); // Make sure item is not checked out
        }


        public string CallNumber
        {
            // Precondition:  None
            // Postcondition: The call number has been returned
            get;
```

```
    // Precondition:  None
    // Postcondition: The call number has been set to the specified value
    set;
}

public string Title
{
    // Precondition:  None
    // Postcondition: The title has been returned
    get;

    // Precondition:  None
    // Postcondition: The title has been set to the specified value
    set;
}

public string Publisher
{
    // Precondition:  None
    // Postcondition: The publisher has been returned
    get;

    // Precondition:  None
    // Postcondition: The publisher has been set to the specified value
    set;
}

public int CopyrightYear
{
    // Precondition:  None
    // Postcondition: The copyright year has been returned
    get
    {
        return itemCopyrightYear;
    }

    // Precondition:  value >= 0
    // Postcondition: The copyright year has been set to the specified value
    set
    {
        if (value >= 0)
            itemCopyrightYear = value;
        else
            throw new ArgumentOutOfRangeException("CopyrightYear",
                value, "CopyrightYear must be >= 0");
    }
}

public int LoanPeriod
{
    // Precondition:  None
    // Postcondition: The loan period has been returned
    get
    {
        return itemLoanPeriod;
    }

    // Precondition:  value >= 0
    // Postcondition: The loan period has been set to the specified value
    set
```

```
    {
        if (value >= 0)
            itemLoanPeriod = value;
        else
            throw new ArgumentOutOfRangeException("LoanPeriod",
                value, "LoanPeriod must be >= 0");
    }
}

// Abstract method header
// Precondition:  daysLate >= 0
// Postcondition: The fee for returning the item the specified days late
//                has been returned
public abstract decimal CalcLateFee(int daysLate);

// Precondition:  None
// Postcondition: The item is checked out by thePatron
public void CheckOut(LibraryPatron thePatron)
{
    itemCheckedOut = true;
    patron = thePatron;
}

// Precondition:  None
// Postcondition: The item is not checked out (by any patron)
public void ReturnToShelf()
{
    itemCheckedOut = false;
    patron = null; // No longer associated with anyone
}

// Precondition:  None
// Postcondition: true is returned if the item is checked out,
//                otherwise false is returned
public bool IsCheckedOut()
{
    return itemCheckedOut;
}

// Precondition:  IsCheckedOut() == true
// Postcondition: The patron that has the item checked out is returned
//                (otherwise null)
public LibraryPatron GetPatron()
{
    return patron;
}

// Precondition:  None
// Postcondition: A string is returned presenting the libary item's data on
//                separate lines
public override string ToString()
{
    String result; // Holds for formatted results as being built

    result = String.Format("Title:\t\t{0}{5}Publisher:\t{1}{5}" +
        "Copyright:\t{2}{5}Loan Period:\t{3}{5}Call Number:\t{4}{5}",
        Title, Publisher, CopyrightYear, LoanPeriod, CallNumber,
        System.Environment.NewLine);

    if (IsCheckedOut())
        result += String.Format("Checked Out By:{1}{0}", GetPatron(),
```

```
                System.Environment.NewLine);
            else
                result += "Not Checked Out";

            return result;
        }
    }
}
```

Library Book Class

```
// File: LibraryBook.cs
// This file creates a concrete LibraryBook class that adds
// an author to the LibraryItem data.
// LibraryBook IS-A LibraryItem

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace LibraryItems
{
    [Serializable]
    public class LibraryBook : LibraryItem
    {
        public const decimal DAILYLATEFEE = 0.25m; // Book's daily late fee

        // Precondition:  theCopyrightYear >= 0 and theLoanPeriod >= 0
        // Postcondition: The library book has been initialized with the specified
        //                values for title, publisher, copyright year, loan period,
        //                call number, and author. The item is not checked out.
        public LibraryBook(String theTitle, String thePublisher, int theCopyrightYear,
            int theLoanPeriod, String theCallNumber, String theAuthor) :
            base(theTitle, thePublisher, theCopyrightYear, theLoanPeriod, theCallNumber)
        {
            Author = theAuthor;
        }

        public string Author
        {
            // Precondition:  None
            // Postcondition: The author has been returned
            get;
            // Precondition:  None
            // Postcondition: The author has been set to the specified value
            set;
        }

        // Precondition:  daysLate >= 0
        // Postcondition: The fee for returning the item the specified days late
        //                has been returned
        public override decimal CalcLateFee(int daysLate)
        {
            if (daysLate >= 0)
                return daysLate * DAILYLATEFEE;
            else
                throw new ArgumentOutOfRangeException("daysLate",
                    daysLate, "daysLate must be >= 0");
```

```
        }

        // Precondition:  None
        // Postcondition: A string is returned presenting the libary item's data on
        //                separate lines
        public override string ToString()
        {
            String result; // Holds for formatted results as being built

            result = String.Format("LibraryBook{2}Author:\t\t{0}{2}{1}",
                Author, base.ToString(), System.Environment.NewLine);

            return result;
        }
    }
}
```

Library Patron Class

```
// File: LibraryPatron.cs
// This file creates a simple LibraryPatron class capable of tracking
// the patron's name and ID.

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace LibraryItems
{
    [Serializable]
    public class LibraryPatron
    {
        private String patronName; // Name of the patron
        private String patronID;   // ID of the patron

        // Precondition:  None
        // Postcondition: The patron has been initialized with the specified name
        //                and ID
        public LibraryPatron(String name, String id)
        {
            PatronName = name;
            PatronID = id;
        }

        public String PatronName
        {
            // Precondition:  None
            // Postcondition: The patron's name has been returned
            get
            {
                return patronName;
            }

            // Precondition:  None
            // Postcondition: The patron's name has been set to the specified value
            set
            {
                patronName = value;
```

```
            }
        }

        public String PatronID
        {
            // Precondition:  None
            // Postcondition: The patron's ID has been returned
            get
            {
                return patronID;
            }

            // Precondition:  None
            // Postcondition: The patron's ID has been set to the specified value
            set
            {
                patronID = value;
            }
        }

        // Precondition:  None
        // Postcondition: A string is returned presenting the library patron's data on
        //                separate lines
        public override string ToString()
        {
            return String.Format("Name:\t\t{0}{2}ID:\t\t{1}",
                PatronName, PatronID, System.Environment.NewLine);
        }

    }
}
```

Library Journal Class

```
// File: LibraryJournal.cs
// This file creates a concrete LibraryJournal class that adds
// discipline and editor.
// LibraryJournal IS-A LibraryPeriodical

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace LibraryItems
{
    [Serializable]
    public class LibraryJournal : LibraryPeriodical
    {
        public const decimal DAILYLATEFEE = 0.75m; // Journal's daily late fee

        // Precondition:  theCopyrightYear >= 0 and theLoanPeriod >= 0 and
        //                theVolume >= 0 and theNumber >= 0
        // Postcondition: The journal has been initialized with the specified
        //                values for title, publisher, copyright year, loan period,
        //                call number, volume, number, discipline, and editor. The
        //                item is not checked out.
        public LibraryJournal(String theTitle, String thePublisher, int theCopyrightYear,
```

```
            int theLoanPeriod, String theCallNumber, int theVolume, int theNumber,
            String theDiscipline, String theEditor) :
            base(theTitle, thePublisher, theCopyrightYear, theLoanPeriod, theCallNumber, theVolume,
theNumber)
        {
            Discipline = theDiscipline;
            Editor = theEditor;
        }

        public string Discipline
        {
            // Precondition:  None
            // Postcondition: The discipline has been returned
            get;

            // Precondition:  None
            // Postcondition: The discipline has been set to the specified value
            set;
        }

        public string Editor
        {
            // Precondition:  None
            // Postcondition: The editor has been returned
            get;

            // Precondition:  None
            // Postcondition: The editor has been set to the specified value
            set;
        }

        // Precondition:  daysLate >= 0
        // Postcondition: The fee for returning the item the specified days late
        //                has been returned
        public override decimal CalcLateFee(int daysLate)
        {
            if (daysLate >= 0)
                return daysLate * DAILYLATEFEE;
            else
                throw new ArgumentOutOfRangeException("daysLate",
                    daysLate, "daysLate must be >= 0");
        }

        // Precondition:  None
        // Postcondition: A string is returned presenting the libary item's data on
        //                separate lines
        public override string ToString()
        {
            String result; // Holds for formatted results as being built

            result = String.Format("LibraryJournal{3}Discipline:\t{0}{3}Editor:\t\t{1}{3}{2}",
                Discipline, Editor, base.ToString(), System.Environment.NewLine);

            return result;
        }
    }
}
```

Library Magazine Class

```
// File: LibraryMagazine.cs
// This file creates a concrete LibraryMagazine class that adds
// no new data.
// LibraryMagazine IS-A LibraryPeriodical

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace LibraryItems
{
    [Serializable]
    public class LibraryMagazine : LibraryPeriodical
    {
        public const decimal DAILYLATEFEE = 0.25m; // Magazine's daily late fee
        public const decimal MAXFEE = 20.00m;      // Max late fee

        // Precondition:  theCopyrightYear >= 0 and theLoanPeriod >= 0 and
        //                theVolume >= 0 and theNumber >= 0
        // Postcondition: The magazine has been initialized with the specified
        //                values for title, publisher, copyright year, loan period,
        //                call number, volume, and number. The item is not checked out.
        public LibraryMagazine(String theTitle, String thePublisher, int theCopyrightYear,
            int theLoanPeriod, String theCallNumber, int theVolume, int theNumber) :
            base(theTitle, thePublisher, theCopyrightYear, theLoanPeriod, theCallNumber, theVolume,
theNumber)
        {
            // No new data to initialize
        }

        // Precondition:  daysLate >= 0
        // Postcondition: The fee for returning the item the specified days late
        //                has been returned
        public override decimal CalcLateFee(int daysLate)
        {
            decimal lateFee = 0.0M; // Late magazine fee

            if (daysLate < 0)
                throw new ArgumentOutOfRangeException("daysLate",
                    daysLate, "daysLate must be >= 0");
            else
                lateFee = daysLate * DAILYLATEFEE;

            // Make sure to cap the late fee
            return Math.Min(lateFee, MAXFEE);
        }

        // Precondition:  None
        // Postcondition: A string is returned presenting the libary item's data on
        //                separate lines
        public override string ToString()
        {
            String result; // Holds for formatted results as being built

            result = String.Format("LibraryMagazine{1}{0}",
                base.ToString(), System.Environment.NewLine);
```

```
            return result;
        }
    }
}
```

Library Movie Class

```
// File: LibraryMovie.cs
// This file creates a concrete LibraryMovie class that adds
// director and rating.
// LibraryMovie IS-A LibraryMediaItem

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace LibraryItems
{
    [Serializable]
    public class LibraryMovie : LibraryMediaItem
    {
        public const decimal DAILYLATEFEEDVD = 1.00m; // DVD/VHS's daily late fee
        public const decimal DAILYLATEFEEBLU = 1.50m; // BluRay's daily late fee
        public const decimal MAXFEE = 25.00m;         // Max late fee

        public enum MPAARatings { G, PG, PG13, R, NC17, U }; // Possible movie ratings

        // String names of ratings - Must put in same order as items in MPAARatings enum
        private String[] RatingsNames = { "G", "PG", "PG-13", "R", "NC-17", "U"};

        private MediaType movieMedium; // The movie's medium

        // Precondition:  theCopyrightYear >= 0 and theLoanPeriod >= 0 and
        //                theMedium from { DVD, BLURAY, VHS } and theDuration >= 0
        // Postcondition: The movie has been initialized with the specified
        //                values for title, publisher, copyright year, loan period,
        //                call number, duration, director, medium, and rating. The
        //                item is not checked out.
        public LibraryMovie(String theTitle, String thePublisher, int theCopyrightYear,
            int theLoanPeriod, String theCallNumber, double theDuration, String theDirector,
            MediaType theMedium, MPAARatings theRating) :
            base(theTitle, thePublisher, theCopyrightYear, theLoanPeriod, theCallNumber, theDuration)
        {
            Director = theDirector;
            Medium = theMedium;
            Rating = theRating;
        }

        public string Director
        {
            // Precondition:  None
            // Postcondition: The director has been returned
            get;

            // Precondition:  None
            // Postcondition: The director has been set to the specified value
            set;
```

```
}

public MPAARatings Rating
{
    // Precondition:  None
    // Postcondition: The rating has been returned
    get;

    // Precondition:  None
    // Postcondition: The rating has been set to the specified value
    set;
}

public override MediaType Medium
{
    // Precondition:  None
    // Postcondition: The medium has been returned
    get
    {
        return movieMedium;
    }

    // Precondition:  value from { DVD, BLURAY, VHS }
    // Postcondition: The medium has been set to the specified value
    set
    {
        if (value == MediaType.BLURAY || value == MediaType.DVD ||
            value == MediaType.VHS)
            movieMedium = value;
        else
            throw new ArgumentOutOfRangeException("Medium",
                value, "Medium must be from { DVD, BLURAY, VHS }");
    }
}

// Precondition:  daysLate >= 0
// Postcondition: The fee for returning the item the specified days late
//                has been returned
public override decimal CalcLateFee(int daysLate)
{
    decimal lateFee = 0.0M; // Late movie fee

    if (daysLate < 0)
        throw new ArgumentOutOfRangeException("daysLate",
            daysLate, "daysLate must be >= 0");
    else if (Medium == MediaType.BLURAY)
        lateFee = daysLate * DAILYLATEFEEBLU;
    else
        lateFee = daysLate * DAILYLATEFEEDVD;

    // Make sure to cap the late fee
    return Math.Min(lateFee, MAXFEE);
}

// Precondition:  None
// Postcondition: A string is returned presenting the libary item's data on
//                separate lines
public override string ToString()
{
    String result; // Holds for formatted results as being built
```

```
            result = String.Format("LibraryMovie{1}Director:\t{0}{1}",
                Director, System.Environment.NewLine);

            result += String.Format("Rating:\t\t{0}{2}{1}",
                RatingsNames[(int)Rating], base.ToString(), System.Environment.NewLine);

            return result;
        }
    }
}
```

Library Music Class

```
// File: LibraryMusic.cs
// This file creates a concrete LibraryMusic class that adds
// artist and number of tracks.
// LibraryMusic IS-A LibraryMediaItem

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace LibraryItems
{
    [Serializable]
    public class LibraryMusic : LibraryMediaItem
    {
        public const decimal DAILYLATEFEE = 0.50m; // Music's daily late fee
        public const decimal MAXFEE = 20.00m;      // Max late fee

        private int numTracks;         // Music's number of tracks
        private MediaType musicMedium; // The movie's medium

        // Precondition:  theCopyrightYear >= 0 and theLoanPeriod >= 0 and
        //                theMedium from { CD, SACD, VINYL } and theDuration >= 0 and
        //                theNumTracks >= 0
        // Postcondition: The movie has been initialized with the specified
        //                values for title, publisher, copyright year, loan period,
        //                call number, duration, director, medium, and rating. The
        //                item is not checked out.
        public LibraryMusic(String theTitle, String thePublisher, int theCopyrightYear,
            int theLoanPeriod, String theCallNumber, double theDuration, String theArtist,
            MediaType theMedium, int theNumTracks) :
            base(theTitle, thePublisher, theCopyrightYear, theLoanPeriod, theCallNumber, theDuration)
        {
            Artist = theArtist;
            Medium = theMedium;
            NumTracks = theNumTracks;
        }

        public string Artist
        {
            // Precondition:  None
            // Postcondition: The artist has been returned
            get;

            // Precondition:  None
            // Postcondition: The artist has been set to the specified value
```

```
    set;
}

public int NumTracks
{
    // Precondition:  None
    // Postcondition: The number of tracks has been returned
    get
    {
        return numTracks;
    }

    // Precondition:  value >= 0
    // Postcondition: The number of tracks has been set to the specified value
    set
    {
        if (value >= 0)
            numTracks = value;
        else
            throw new ArgumentOutOfRangeException("NumTracks",
                value, "NumTracks must be >= 0");
    }
}

public override MediaType Medium
{
    // Precondition:  None
    // Postcondition: The medium has been returned
    get
    {
        return musicMedium;
    }

    // Precondition:  value from { CD, SACD, VINYL }
    // Postcondition: The medium has been set to the specified value
    set
    {
        if (value == MediaType.CD || value == MediaType.SACD ||
            value == MediaType.VINYL)
            musicMedium = value;
        else
            throw new ArgumentOutOfRangeException("Medium",
                value, "Medium must be from { CD, SACD, VINYL }");
    }
}

// Precondition:  daysLate >= 0
// Postcondition: The fee for returning the item the specified days late
//                has been returned
public override decimal CalcLateFee(int daysLate)
{
    decimal lateFee = 0.0M; // Late music fee

    if (daysLate < 0)
        throw new ArgumentOutOfRangeException("daysLate",
            daysLate, "daysLate must be >= 0");
    else
        lateFee = daysLate * DAILYLATEFEE;

    // Make sure to cap the late fee
    return Math.Min(lateFee, MAXFEE);
```

```
        }

        // Precondition:  None
        // Postcondition: A string is returned presenting the libary item's data on
        //                separate lines
        public override string ToString()
        {
            String result; // Holds for formatted results as being built

            result = String.Format("LibraryMusic{3}Artist:\t\t{0}{3}Num Tracks:\t{1}{3}{2}",
                Artist, NumTracks, base.ToString(),System.Environment.NewLine);

            return result;
        }
    }
}
```

Library Periodical Class

```
// File: LibraryPeriodical.cs
// This file creates an abstract LibraryPeriodical class that adds
// volume and number.
// LibraryPeriodical IS-A LibraryItem

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace LibraryItems
{
    [Serializable]
    public abstract class LibraryPeriodical : LibraryItem
    {
        private int periodicalVolume; // The periodical's volume
        private int periodicalNumber; // The preiodical's number

        // Precondition:  theCopyrightYear >= 0 and theLoanPeriod >= 0 and
        //                theVolume >= 0 and theNumber >= 0
        // Postcondition: The library periodical has been initialized with the specified
        //                values for title, publisher, copyright year, loan period,
        //                call number, volume, and number. The item is not checked out.
        public LibraryPeriodical(String theTitle, String thePublisher, int theCopyrightYear,
            int theLoanPeriod, String theCallNumber, int theVolume, int theNumber) :
            base(theTitle, thePublisher, theCopyrightYear, theLoanPeriod, theCallNumber)
        {
            Volume = theVolume;
            Number = theNumber;
        }

        public int Volume
        {
            // Precondition:  None
            // Postcondition: The volume has been returned
            get
            {
                return periodicalVolume;
            }
```

```
            // Precondition:  value >= 0
            // Postcondition: The volume has been set to the specified value
            set
            {
                if (value >= 0)
                    periodicalVolume = value;
                else
                    throw new ArgumentOutOfRangeException("Volume",
                        value, "Volume must be >= 0");
            }
        }

        public int Number
        {
            // Precondition:  None
            // Postcondition: The number has been returned
            get
            {
                return periodicalNumber;
            }

            // Precondition:  value >= 0
            // Postcondition: The number has been set to the specified value
            set
            {
                if (value >= 0)
                    periodicalNumber = value;
                else
                    throw new ArgumentOutOfRangeException("Number",
                        value, "Number must be >= 0");
            }
        }

        // Precondition:  None
        // Postcondition: A string is returned presenting the libary item's data on
        //                separate lines
        public override string ToString()
        {
            String result; // Holds for formatted results as being built

            result = String.Format("Volume:\t\t{0}{3}Number:\t\t{1}{3}{2}",
                Volume, Number, base.ToString(), System.Environment.NewLine);

            return result;
        }
    }
}
```

Library Media Item Class

```
// File: LibraryMediaItem.cs
// This file creates an abstract LibraryMediaItem class that adds
// media type and duration.
// LibraryMediaItem IS-A LibraryItem

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
```

```
namespace LibraryItems
{
    [Serializable]
    public abstract class LibraryMediaItem : LibraryItem
    {
        public enum MediaType { DVD, BLURAY, VHS, CD, SACD, VINYL }; // Possible media types

        // String names of media types - Must put in same order as items in MediaType enum
        private String[] MediaTypeNames = { "DVD", "Blu-Ray", "VHS", "CD", "SACD", "VINYL" };

        private double itemDuration; // The item's duration (in minutes)

        // Precondition:  theCopyrightYear >= 0 and theLoanPeriod >= 0 and
        //                theDuration >= 0
        // Postcondition: The library media item has been initialized with the specified
        //                values for title, publisher, copyright year, loan period,
        //                call number, and duration. The item is not checked out.
        public LibraryMediaItem(String theTitle, String thePublisher, int theCopyrightYear,
            int theLoanPeriod, String theCallNumber, double theDuration) :
            base(theTitle, thePublisher, theCopyrightYear, theLoanPeriod, theCallNumber)
        {
            Duration = theDuration;
        }

        // Abstract property header
        public abstract MediaType Medium
        {
            // Precondition:  None
            // Postcondition: The medium has been returned
            get;

            // Precondition:  Varies - See concrete implementation
            // Postcondition: The medium has been set to the specified value
            set;
        }

        public double Duration
        {
            // Precondition:  None
            // Postcondition: The duration has been returned
            get
            {
                return itemDuration;
            }

            // Precondition:  value >= 0
            // Postcondition: The duration has been set to the specified value
            set
            {
                if (value >= 0)
                    itemDuration = value;
                else
                    throw new ArgumentOutOfRangeException("ItemDuration",
                        value, "ItemDuration must be >= 0");
            }
        }

        // Precondition:  None
        // Postcondition: A string is returned presenting the libary item's data on
        //                separate lines
```

```
        public override string ToString()
        {
            String result; // Holds for formatted results as being built

            result = String.Format("Duration:\t{0:F1}{1}",
                Duration, System.Environment.NewLine);

            result += String.Format("Medium:\t\t{0}{2}{1}",
                MediaTypeNames[(int)Medium], base.ToString(), System.Environment.NewLine);

            return result;
        }
    }
}
```

APPENDIX B: Legacy Systems

DOS batch file (for CIS 350, infrastructure tech)

DOS batch file

By Charles Rady for CIS 350

```
ECHO OFF
ECHO THIS IS A SIMPLE BATCH FILE FOR LAB 2 BY CHARLES RADY
ECHO STRIKE ANY KEY TO SEE THE DIRECTORY OF DISK (E:)
PAUSE
DIR E:
ECHO SRIKE ANY KEY TO SEE THE DIRECTORY OF YOUR MAIN DISK (C:)
PAUSE
DIR C:
REM THIS IS AN EXERCISE
ECHO CHECKING FOR ENDLESS LOOPS....
IF EXIST E:LOOP.BAT GOTO YES
ECHO THERE ARE NO ENDLESS LOOPS
GOTO NONE
:YES
ECHO THIS 1980'S DISK OPERATING SYSTEM HAS FOUND AN ENDLESS LOOP CALLED
ECHO LOOP.BAT. YOU SHOULD NOT RUN THIS BATCH FILE
GOTO DONE
:NONE
ECHO YOUR SYSTEM IS CLEAN OF ANY ENDLESS LOOPS
:DONE
ECHO MOVING ON.....
ECHO STRIKE ANY KEY TO CHANGE THE DATE
DATE
ECHO STRIKE ANY KEY TO CHANGE THE TIME
TIME
MD TEST1
MD TEST2
DIR E:TEST?
DIR E:TEST*
COPY E:LOOP.BAT E:TEST1
ECHO ONE FILE HAS BEEN COPIED TO THE DIRECTORY: E:\TEST1
RD TEST2
TYPE E:\TEST1\TESTFILE
SORT/R E:\TEST1\TESTFILE
ECHO HERE IS YOUR DOS VERSION
VER
CD E:\TEST1
```

APPENDIX C: CDRadyproductions.com

CDRadyproductions.com was built with a combination of PHP and HTML pages. It contains music, images, and an online portfolio. It was built with Microsoft Expression Web 4 and is hosted by GoDaddy.com.

- **All music on this site was written, performed, recorded, and mixed by Charles Rady**
- **All images (including banners and backgrounds) were originally photographed by Charles Rady, then edited for web-display by Charles Rady**
- **Files in the online portfolio are by Charles Rady (in cases where a group created the file, the other group members are given credit)**
- **All music and images © 2010 Charles Damien Rady**
- **All published writing © 2010 Charles Damien Rady**
- **Unpublished writing has been graded and belongs to Charles Rady**
- **Other assignments: Charles Rady**

APPENDIX D: REFERENCES

Professional:

See provided references (2) from Holland America Line

Semi-Professional

Randa Koczera
267 Scotts Point Rd.
Clifton, Me, 04428
(207)-991-1119
Owner: DollClothesShop.com
Randa@dollclothesshop.com
Consultant

Personal:

Barbara Buchanan
PO Box 197054
Louisville, KY 40259
(502) 964-0315
multicruiser@aol.com
Friend of family, Godmother

David Cornell

David Cornell
PO Box 201
Eddington, ME 04428
cornelld92@yahoo.com
Apartment Owner/Merchant Marine Engineer
*Currently at sea

Holland America Line references:

This review was passed on from the ship's cruise staff to my agent, and then to me.

-----Original Message-----
From: maestroscottm@aol.com
To: markandert@hotmail.com
Sent: Fri, 9 Nov 2007 1:43 pm
Subject: Charley Rady

Mark,

I am a Music Director for Holland America and most recently had the pleasure of working with Charles Rady on board the MS Maasam. I was very impressed with Charley as a Pianist/Entertainer. Holland America is in the process of trying to keep our returning passengers happy and still appeal to a new and younger customer. This is a difficult line to walk, especially in the Piano Bar, where passenger loyalty is necessary for success. Charley came on the ship a little timid, but immediately inquired about passenger requests and working within the style of Holland America. In a matter of days Charley was learning new material and adapting his own style to whatever the passengers and the ship required. He sometimes would learn 10- 20 new songs a day in addition to performing amazing renditions of his stock repertoire! (Elton John, Billy Joel, etc.) He was amazingly tenacious and his hard work payed off. He had a loyal following on every cruise. I was always running into passengers who would give me positive feedback about Charley. He is talented, hard working and a great personality to have on board. He makes the ship a better place to be.

Scott McKenzie

Music director

Holland America

Holland America Line: Actual evaluation

Holland America Line

A Signature of Excellence

Entertainment Staff Evaluation

Charles Rady – 63999
Pianist (Piano Bar)

1. Dress / Grooming	M
2. Guest Relations	M
3. Attitude / Professionalism	E
4. Initiative / Motivation / Organization	M
5. Hosting Activities (CS only)	
6. Musicianship (Musicians only)	E
7. Environmental Compliance	M

E – Exceeds Expectations **M – Meets Expectations** **D – Does Not Meet Expectations**

COMMENTS:

- Charlie is a great musician with excellent musical knowledge.
- Communicates well with CD and EM at all times.
- Always on time for scheduled sets.
- Reliable and professional at all times. Is always willing to help out when needed.
- Continues to receive positive written and verbal comments on his musicianship and showmanship.
- Has an enthusiastic and positive attitude with guests, co-workers and supervisors.
- Charlie has used his creativity to program and implement new events in the Piano Bar. The attendance has improved dramatically with the additions of his new themes.
- Charlie is always open to suggestions and is always looking for areas to improve and boost attendance in his lounge.
- Charlie is a wonderful musician and has a great attitude all the time! He is a great asset to the company and we would love to have him return to the Volendam after dry dock.

Cruise Director Rebecca Fabischek

Event Manager Armando Perez

Ship Volendam

Date 10 April 2008

Employee Charles Rady

☐ 1 MONTH
☐ MID TERM
☒ FINAL